UNIVERSIDAD NACIONAL
"PEDRO RUIZ GALLO"
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
ESCUELA PROFESIONAL DE MATEMÁTICA

Localización y corrección de errores en la transmisión de mensajes mediante códigos BCH

Tesis

Para optar el Título Profesional de

Licenciado en Matemáticas

presentado por:

Bach. Mat. Eduardo Enrique Oliva Suárez

Asesor

Dr. Lic. Mat. Walter Arriaga Delgado
ORCID: 0000-0001-9311-5314

Lambayeque – Perú

Marzo – 2023

Universidad Nacional Pedro Ruiz Gallo

Facultad de Ciencias Físicas y Matemáticas

Escuela Profesional de Matemática

"Localización y corrección de errores en la transmisión de mensajes mediante códigos BCH"

Dr. Lic. Mat. Walter Arriaga Delgado

Asesor

Bach. Mat. Eduardo Enrique Oliva Suárez

Autor

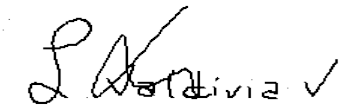
Lambayeque – Perú

Marzo – 2023

Universidad Nacional Pedro Ruiz Gallo

Escuela Profesional De Matemática

Los firmantes, por la presente certifican que han leído y recomiendan a la Facultad de Ciencias Físicas y Matemáticas la aceptación de la tesis titulada “Localización y corrección de errores en la transmisión de mensajes mediante códigos BCH”, presentado por el Bach. Mat. Eduardo Enrique Oliva Suárez en el cumplimiento parcial de los requisitos necesarios para la obtención del título profesional de Licenciado en Matemáticas.



Dr. Lic. Mat. Segundo Leonardo Valdivia Velásquez

Jurado Presidente

Dr. Lic. Mat. Rubén Esteban Burga Barboza

Jurado Secretario

Lic. Mat. Miguel Angel Baca Ferreyros

Jurado Vocal

Fecha de defensa

Marzo – 2023

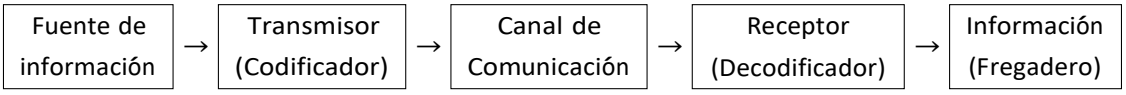
Índice general

Introducción

La teoría de codificación estudia los métodos para lograr una transmisión eficiente y exacta de mensajes desde un lugar a otro, la cual ha sido desarrollada para diversas aplicaciones como: la minimización de ruido en la grabación de un disco compacto, transferencia de datos a partir de un ordenador a otro, o de una memoria USB a la unidad central de proceso (CPU) y la emisión de información de una distante fuente, tal como un satélite del tiempo o de comunicaciones, o la nave espacial Voyager que enviaron cuadros de Júpiter y de Saturno a la tierra.

El medio físico con el cual se transmite la información se llama **canal**, los circuitos telefónicos y el ambiente son ejemplos más comunes de canal, entre otros. Los disturbios indeseables, conocidos como **ruido**, tienden a causar que se reciba un mensaje distinto a lo enviado. El **ruido** es causado por manchas solares, relámpagos, dobleces de una cinta magnética, lluvia de meteoritos, mensaje telefónico competente, disturbio de radio al azar, mecanografiar pobre, audiencia pobre, entre otras cosas.

La teoría de codificación, se encarga con el problema de localizar y reparar los errores de una transmisión de mensaje, provocados por el ruido. El siguiente diagrama proporciona una idea aproximada de un sistema de transmisión de mensajes de forma general:



La parte más importante del diagrama, por lo que nos referimos, es el ruido, ya que por ello es el desarrollo de esta teoría.

En lo habitual para controlar el ruido, la opción que se tiene es utilizar un buen canal para emitir mensajes, además del uso de varios filtros antiruidos y de esa manera contender ciertos tipos de interferencia que puedan ser detectados; de estas preocupaciones es de lo que se encarga la ingeniería.

Cuando ya se encuentra ubicado un buen sistema mecánico para la solución de estos problemas, nos concentramos en la construcción del decodificador, de tal manera que cumpla lo siguiente:

1. Pronta codificación del mensaje.

2. Transmisión factible del mensaje codificado.
3. El decodificado rápido de mensaje recibido.
4. Reparar errores ingresados en el canal.
5. Traslado máximo del mensaje por unidad de tiempo.

El objetivo principal es el cuarto, teniendo en cuenta que no siempre es compatible con el primero; además que en casos especiales no lo es con los otros tres. Cualquiera que sea una solución, es necesario buscar una comprensión entre los cinco objetivos. Diariamente cuando nos comunicamos, entre una u otras estándar, utilizamos las palabras habladas o escritas formados por un abecedario definido. Si se quiere comunicar una información, se codifica en cadenas de palabras que después escribimos o hablamos; ésta se envía a través de un canal; el canal normalmente es el espacio de la boca al oído o de la pluma al papel.

Las causas que ocasionan el ruido puede ser: discurso pobre, mala audiencia, gramática incorrecta, discurso estéreo, competencia ruidosa, falta de ortografía, mala lectura, o una máquina de escribir culpable. Nuestra lectura (audiencia) y la comprensión de la información recibida sería el decodificador.

El método estándar para combatir errores es con redundancia. Muchos negocios en la actualidad agrega un dígito de verificación para identificar números, estos son dígitos extra que son usados para verificar la corrección de datos o de los números de cuenta. Probablemente este sea el método más común de codificación en la vida cotidiana. Nos ocuparemos de ideas más sofisticadas pero muy similares.

Capítulo 1

Códigos Lineales Cíclicos

1.1 Asunciones Básicas

Indicamos algunas definiciones y asunciones fundamentales que se aplican a través de la tesis.

Definición 1.1.

1. La información a enviar, se envía en una secuencia de 0 y 1, llamados *dígitos*.
2. La secuencia de dígitos formada se llama palabra.
3. Dada una palabra, su longitud es el número de sus dígitos. Por ejemplo, 01101010 es una palabra de longitud ocho.
4. Una palabra se envía, enviando los dígitos, uno por uno por un *canal binario*, refiriéndose al hecho que solo se utilizan dos dígitos, 0 y 1.

Cada dígito es enviado de manera mecánica, eléctrica, magnética, o de otra manera por uno de dos tipos de pulsos fácilmente distinguidos.

Definición 1.2. Un conjunto $\mathcal{C}$ conformado por palabras, llamaremos *código binario*.

Todo código formado por palabras de longitud dos es:

$$\mathcal{C} = \{00, 10, 01, 11\}$$

Definición 1.3. Dígase que un código es de bloque (**código de bloque**), cuando es un código formado por todas las palabras que tienen la misma longitud; a este número se le llama longitud de un código. Consideraremos solamente códigos de bloque. Así pues, el término **código** significará siempre un **código de bloque binario**.

Definición 1.4. Las palabras que son elementos de un código dado $\mathcal{C}_0$, se llamarán palabras−código. Denotaremos por $|\mathcal{C}|$, al número de palabras - código de un código $\mathcal{C}$.

También necesitamos hacer ciertas asunciones básicas sobre el canal. Estas asunciones formarán necesariamente la teoría que formulamos.

Asunción 1. Las palabras - código de longitud *n* que consisten en $0's$ y $1's$ son recibidos como una palabra de longitud *n* que consiste en $0's$ y $1's$, aunque no necesariamente igual que la palabra que fue enviada.

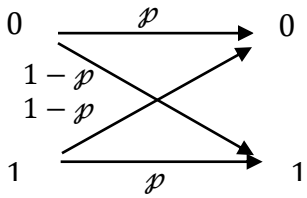
Asunción 2. De la primera palabra transmitida determine el comienzo. Entonces, si usamos palabras código de longitud 3 y recibimos 011011001, sabemos que las palabras recibidas son, en el orden, 011, 011, 001. Por medio de esta asunción, usando otra vez la longitud 3, el canal no puede entregar la palabra 01101 al receptor, porque se ha perdido un dígito aquí.

Asunción 3. El ruido está dispersado aleatoriamente en comparación con estar en los grupos de ruido llamados explosiones. Es decir, la probabilidad de cualquier dígito que es afectado en su envío, es igual que la de cualquier otro y no es influenciada por los errores hechos en dígitos vecinos. Esta asunción no es muy realista para muchos tipos de ruido, como por ejemplo: relámpagos o rasguños en un disco compacto, lo cual consideraremos eventual este tipo de ruido.

Cuando se tiene un canal sin un tipo de ruido o silencioso (*canal perfecto*), si se envía los dígitos 0 o 1, se recibe con certeza los mismos dígitos. Si tuviéramos siempre un canal perfecto, la teoría de códigos no tendría mucha importancia. Para nuestro interés (aunque no para otros) no existe un canal perfecto, todos los canales son ruidosos; existen canales que son más confiables por ser menos ruidosos que otros.

Se dice que un canal binario es simétrico si los dígitos 0 y 1 se transmiten con la misma precisión; es decir, la posibilidad de obtener el dígito correcto no depende del dígito 0 ó 1 del cual se envió. Un número real p, $0 \leq p \leq 1$, se le llama confiabilidad de un canal binario (CBS), donde el número transmitido sea el número recibido es la probabilidad del número real p.

De que el número que se recibe sea igual al número enviado es la probabilidad de p, podemos decir que la probabilidad de que el número recibido no sea el número enviado es $1 - p$. Este diagrama puede ayuda a comprender cómo funciona un CBS:



Se puede complicar al evaluar el valor p real para un canal en particular en la mayoría de los casos. Pero la forma de la teoría no es afectada por el valor que pueda tomar p.

Si un canal es más confiable, decimos que es más confiable que otro. Recuerda que si $p = 1$, los números no pueden cambiar durante la transmisión, por lo que el canal será perfecto y lo cual no nos interesa. Si $p = 0$, no sería un canal; y si consideramos un canal con $0 < p \leq 1/2$ se convertiría fácilmente en un canal con $1/2 \leq p < 1$. *Por lo tanto en adelante asumiremos que estamos utilizando un CBS con la probabilidad p satisfaciendo* $1/2 < p < 1$.

Tenemos $\mathcal{K} = \{0, 1\}$ y además $\mathcal{K}^n$, todas las palabras binarias en conjunto con longitud *n*. La adición y producto de elementos de $\mathcal{K}$ se define:

- $0 + 0 = 0$, $1 + 1 = 0$, $0 + 1 = 1$, $1 + 0 = 1$
- $0 \cdot 0 = 0$, $1 \cdot 1 = 1$, $0 \cdot 1 = 0$, $1 \cdot 0 = 0$

La adición para elementos de $\mathcal{K}^n$, se define usando la adición sobre $\mathcal{K}$ para cada elemento. Así, por ejemplo, sean las palabras *u, v* donde:

$u = 011100$ y $v = 111001$, entonces $u + v = 100101$.

Claramente, agregar palabras binarias de longitud *n,* se tiene como resultado palabra binaria de longitud *n*, por lo que $\mathcal{K}^n$ bajo la adición es cerrado.

Usando expresiones de algebra lineal, denotamos los elementos de $\mathcal{K}$ como escalares; por lo tanto, en $\mathcal{K}^n$ la multiplicación escalar se definido elemento por elemento. Como los escalares son solamente 0 y 1, en una palabra u los únicos múltiplos serían $0 \cdot u$, que pertenece a $\mathcal{K}^n$ con 0 en todos sus elementos, y $1 \cdot u$, el cuál es u. Llamamos al elemento de $\mathcal{K}^n$ donde todos sus componentes son 0 como *la palabra cero*. Está claro que $\mathcal{K}^n$ es cerrado bajo el producto escalar.

Usando estas definiciones de adición y producto escalar, se puede demostrar que $\mathcal{K}^n$ es un espacio vectorial. Esto es, sea cualesquiera palabras r, s y t de longitud n, además sean a y b escalares arbitrarios:

i. $s + t \in \mathcal{K}^n$

ii. $(r + s) + t = r + (s + t)$

iii. $s + 0 = 0 + s$, donde 0 es la palabra cero.

iv. Para algún $s' \in \mathcal{K}^n$, $s + s' = s' + s = 0$

v. $s + t = t + s$

vi. $as \in \mathcal{K}^n$

vii. $a(s + t) = as + at$

viii. $(a + b)s = as + bs$

ix. $(ab)s = a(bs)$

x. $1s = s$

Demostración.

i. $s + t \in \mathcal{K}^n$

En efecto:

Sea $s = (s_1, s_2, \ldots, s_n)$ y $t = (t_1, t_2, \ldots, t_n)$; tomemos $r = s + t$, $r = (r_1, r_2, \ldots, r_n)$ de esto tenemos que:

$$r_i = \begin{cases} 1 & si\ s_i \neq t_i \\ 0 & si\ s_i = t_i \end{cases}$$

lo que nos indica que $s + t \in \mathcal{K}^n$

ii. $(r + s) + t = r + (s + t)$.

En efecto:

Sea $r = (r_1, r_2, \ldots, r_n)$, $s = (s_1, s_2, \ldots, s_n)$ y $t = (t_1, t_2, \ldots, t_n)$

$$\begin{aligned}
(r + s) + t &= [(r_1, r_2, \ldots, r_n) + (s_1, s_2, \ldots, s_n)] + (t_1, t_2, \ldots, t_n) \\
&= (r_1 + s_1, r_2 + s_2, \ldots, r_n + s_n) + (t_1, t_2, \ldots, t_n) \\
&= ((r_1 + s_1) + t_1, (r_2 + s_2) + t_2, \ldots, (r_n + s_n) + t_n) \\
&= (r_1 + (s_1 + t_1), r_2 + (s_2 + t_2), \ldots, r_n + (s_n + t_n)) \quad \text{asociando} \\
&= (r_1, r_2, \ldots, r_n) + (s_1 + t_1, s_2 + t_2, \ldots, s_n + t_n) \\
&= (r_1, r_2, \ldots, r_n) + [(s_1, s_2, \ldots, s_n) + (t_1, t_2, \ldots, t_n)] \\
&= r + (s + t)
\end{aligned}$$

Tomando extremos tenemos que $(r + s) + t = r + (s + t)$

iii. $s + 0 = 0 + s$, donde 0 es la palabra cero.

En efecto:

Sea $s = (s_1, s_2, \ldots, s_n)$ y $0 = (0, 0, \ldots, 0)$

$$\begin{aligned} s + 0 &= (s_1, s_2, \ldots, s_n) + (0, 0, \ldots, 0) \\ &= (s_1 + 0, s_2 + 0, \ldots, s_n + 0) \\ &= (0 + s_1, 0 + s_2, \ldots, 0 + s_n) \qquad \text{conmutando} \\ &= (0, 0, \ldots, 0) + (s_1, s_2, \ldots, s_n) \\ &= 0 + s \end{aligned}$$

Tomando extremos tenemos que $s + 0 = 0 + s$

iv. Para algún $s' \in \mathcal{K}^n$, $s + s' = s' + s = 0$

En efecto:

Si $s + s' = 0$ entonces s y s' no difieren en algún lugar; esto quiere decir que $s' = s$.

Si $s' + s = 0$ entonces s' y s no difieren en algún lugar; esto quiere decir que $s' = s$.

v. $s + t = t + s$

En efecto:

Sea $s = (s_1, s_2, \ldots, s_n)$ y $t = (t_1, t_2, \ldots, t_n)$

$$\begin{aligned} s + t &= (s_1, s_2, \ldots, s_n) + (t_1, t_2, \ldots, t_n) \\ &= (s_1 + t_1, s_2 + t_2, \ldots, s_n + t_n) \\ &= (t_1 + s_1, t_2 + s_2, \ldots, t_n + s_n) \qquad \text{conmutando} \\ &= (t_1, t_2, \ldots, t_n) + (s_1, s_2, \ldots, s_n) \\ &= t + s \end{aligned}$$

Tomando extremos tenemos que $s + t = t + s$

vi. $as \in \mathcal{K}^n$

En efecto:

Sea $s = (s_1, s_2, \ldots, s_n)$, entonces

$$\begin{aligned} as &= a(s_1, s_2, \ldots, s_n) \\ &= (as_1, as_2, \ldots, as_n) \\ &= (t_1, t_2, \ldots, t_n) \\ &= t \in \mathcal{K}^n \end{aligned}$$

Tomando extremos tenemos que $as \in \mathcal{K}^n$.

vii. $a(s + t) = as + at$

En efecto:

Sea $s = (s_1, s_2, \ldots, s_n)$ y $t = (t_1, t_2, \ldots, t_n)$, entonces

$$\begin{aligned} a(s + t) &= a[(s_1, s_2, \ldots, s_n) + (t_1, t_2, \ldots, t_n)] \\ &= a(s_1 + t_1, s_2 + t_2, \ldots, s_n + t_n) \\ &= (a(s_1 + t_1), a(s_2 + t_2), \ldots, a(s_n + t_n)) \\ &= (as_1 + at_1, as_2 + at_2, \ldots, as_n + at_n) \\ &= (as_1, as_2, \ldots, as_n) + (at_1, at_2, \ldots, at_n) \\ &= as + at \end{aligned}$$

Tomando extremos tenemos que $a(s + t) = as + at$

viii. $(a + b)s = as + bs$

En efecto:

Sea $s = (s_1, s_2, \ldots, s_n)$, entonces

$$\begin{aligned} (a + b)s &= (a + b)(s_1, s_2, \ldots, s_n) \\ &= ((a + b)s_1, (a + b)s_2, \ldots, (a + b)s_n) \\ &= (as_1 + bs_1, as_2 + bs_2, \ldots, as_n + bs_n)) \\ &= (as_1, as_2, \ldots, as_n) + (bs_1, bs_2, \ldots, bs_n) \\ &= a(s_1, s_2, \ldots, s_n) + b(s_1, s_2, \ldots, s_n) \\ &= as + bs \end{aligned}$$

Tomando extremos tenemos que $(a + b)s = as + bs$

ix. $(ab)s = a(bs)$

En efecto:

Sea $s = (s_1, s_2, \ldots, s_n)$, entonces

$$\begin{aligned} (ab)s &= (ab)(s_1, s_2, \ldots, s_n) \\ &= ((ab)s_1, (ab)s_2, \ldots, (ab)s_n) \\ &= (a(bs_1), a(bs_2), \ldots, a(bs_n)) \qquad \text{asociando} \\ &= a(bs_1, bs_2, \ldots, bs_n) \\ &= a[b(s_1, s_2, \ldots, s_n)] \\ &= a(bs) \end{aligned}$$

Tomando extremos tenemos que $(ab)s = a(bs)$

x. $1s = s$

En efecto:

Sea $s = (s_1, s_2, \ldots, s_n)$, entonces

$$\begin{aligned} 1s &= 1 \cdot (s_1, s_2, \ldots, s_n) \\ &= (1 \cdot s_1, 1 \cdot s_2, \ldots, 1 \cdot s_n) \\ &= (s_1, s_2, \ldots, s_n) \\ &= s \end{aligned}$$

□

1.2 Polinomios y Palabras

Será conveniente para nosotros representar los códigos cíclicos en términos de polinomios. Por esta razón, revisaremos algunos hechos necesarios sobre los polinomios (de una variable).

Decimos que un *polinomio de grado n sobre* $\mathcal{K}$ es un polinomio $a_0 + a_1x + \cdots + a_nx^n$ donde los coeficientes $a_0, \ldots, a_n$ son elementos de $\mathcal{K}$. Todos los polinomios en conjunto sobre $\mathcal{K}$ se denota por $\mathcal{K}[x]$. Los elementos de $\mathcal{K}[x]$ estarán representados por $f(x)$, $g(x)$, $p(x)$, etc.

Los polinomios sobre $\mathcal{K}$ se suman y multiplican de manera antes definidas teniendo en cuenta que como 1 + 1 = 0, se concluye que $x^k + x^k = 0$. Esto significa que el grado de $f(x) + g(x)$ no será el máx(grad $f(x)$, grad $g(x)$) necesariamente.

Ejemplo 1.1. Sea $f(x) = 1 + x + x^3 + x^4$, $g(x) = x + x^2 + x^3$ y $h(x) = 1 + x^2 + x^4$. Entonces

(a) $f(x) + g(x) = 1 + x^2 + x^4$;

(b) $f(x) + h(x) = x + x^2 + x^3$

(c) $f(x).g(x) = (x + x^2 + x^3) + x(x + x^2 + x^3) + x^3(x + x^2 + x^3) + x^4(x + x^2 + x^3)$

$f(x)g(x) = x + x^7$

Para los polinomios sobre $\mathcal{K}$, el desarrollo de división larga habitual funciona igual tal como se hace para polinomios sobre los números racionales.

Definición 1.5 (Algoritmo de la división)**.** Sea $f(x)$ y $h(x)$ en $\mathcal{K}[x]$ con $h(x) \neq 0$. Entonces existe un único polinomio $q(x)$ y $r(x)$ en $\mathcal{K}[x]$ tal que

$$f(x) = q(x)\, h(x) + r(x),$$

con $r(x) = 0$ o $\text{grad}(r(x)) < \text{grad}(h(x))$.

Demostración. Primero demostraremos la existencia de los polinomios $q(x)$ y $r(x)$ y luego probaremos la unicidad de dichos polinomios.
Para construir los polinomios $q(x)$ y $r(x)$ se realiza lo siguiente:

1. Si $f(x) = 0$ entonces $0 = 0.\, h(x) + 0$. Por lo tanto

$$q(x) = 0 \qquad \wedge \qquad r(x) = 0$$

2. Si $f(x) \neq 0$ entonces $\text{grad}\, f(x) = n$; $\text{grad}\, h(x) = m$, se tiene 2 casos

Caso 1: Si $m > n$ entonces $h(x) = 0$

$$\begin{aligned} f(x) &= 0 \cdot q(x) + r(x) \\ f(x) &= r(x) \\ \text{grad}\, r(x) &= \text{grad}\, f(x) < \text{grad}\, h(x) \\ \text{grad}\, f(x) &< \text{grad}\, h(x) \end{aligned}$$

Caso 2: Si $m \leq n$, procederemos por inducción en n. Sean

$$\begin{aligned} f(x) &= a_0 + a_1 x + \cdots + a_n x^n \\ h(x) &= b_0 + b_1 x + \cdots + b_m x^m \end{aligned}$$

Consideremos un polinomio de grado menor que $f(x)$

$$f'(x) = f(x) + \frac{a_n}{b_m} x^{n-m} h(x) \qquad (1)$$

Aplicando la hipótesis inductiva, se tiene:

$$f'(x) = q'(x)h(x) + r(x)$$

donde $r(x) = 0 \vee \text{grad}\, r(x) < \text{grad}\, h(x)$.
Reemplazando (2) en (1), se tiene:

$$f(x) + \frac{a_n}{b_m} x^{n-m} h(x) = q'(x)h(x) + r(x)$$

usando operaciones definidas sobre $\mathcal{K}$, tenemos:

$$f(x) = \left(q'(x) + \frac{a_n}{b_m}x^{n-m}\right)h(x) + r(x)$$

eligiendo $q(x) = q'(x) + \frac{a_n}{a_m}x^{n-m}$, se tiene que:

$$f(x) = q(x)h(x) + r(x)$$

Para demostrar la unicidad, supongamos que existen $q_1(x)$ y $r_1(x)$ tal que:

$$f(x) = q_1(x)h(x) + r_1(x)$$

donde $r_1(x) = 0$ y grad $r_1(x) <$ grad $h(x)$; de manera que:

$$q(x)h(x) + r(x) = q_1(x)h(x) + r_1(x)$$

usando operaciones definidas sobre $\mathcal{K}$,

$$h(x)[q(x) + q_1(x)] + [r(x) + r_1(x)] = 0$$

Por polinomio nulo se tiene:

$$q(x) + q_1(x) = 0 \qquad \wedge \qquad r(x) + r_1(x) = 0$$

si se anulan, entonces

$$q(x) = q_1(x) \qquad \wedge \qquad r(x) = r_1(x)$$

Por lo tanto $q(x)$ y $r(x)$ son únicos.

□

El polinomio $q(x)$ es nombrado *cociente*, y $r(x)$ es nombrado *residuo*. El procedimiento para localizar *cociente* y *residuo* cuando $h(x)$ es divisible en $f(x)$ es el mismo de una división habitual, con la aritmética en $\mathcal{K}$ entre sus coeficientes.

Ejemplo 1.2. Sea $f(x) = x + x^2 + x^6 + x^7 + x^8$ y $h(x) = 1 + x + x^2 + x^4$. Entonces

$$\begin{array}{r|l} & x^4 + x^3 \\ \hline x^4 + x^2 + x + 1 & x^8 + x^7 + x^6 + x^2 + x \\ & \underline{x^8 + x^6 + x^5 + x^4} \\ & x^7 + x^5 + x^4 + x^2 + x \\ & \underline{x^7 + x^5 + x^4 + x^3} \\ & x^3 + x^2 + x \end{array}$$

Así el cociente es $q(x) = x^3 + x^4$ y el residuo es $r(x) = x + x^2 + x^3$. Podemos escribir $f(x) = h(x)(x^3 + x^4) + (x + x^2 + x^3)$. Note que el grad$(r(x))$ < grad$(h(x)) = 4$

El polinomio $f(x) = a_0 + a_1x + a_2x^2 + \cdots + a_{n-1}x^{n-1}$ de grado a lo sumo $n - 1$ sobre $\mathcal{K}$ puede ser considerado como la palabra $u = a_0a_1a_2 \cdots a_{n-1}$ de longitud n en $\mathcal{K}^n$. Por ejemplo si $n = 7$

Polinomio	Palabra
$1 + x + x^2 + x^4$	1110100
$1 + x^4 + x^5 + x^6$	1000111
$1 + x + x^3$	1101000

De esta manera cualquier código $\mathcal{C}$ con longitud n se puede representar sobre $\mathcal{K}$ como un conjunto de polinomios, a lo mucho con grado $n - 1$.

Note que puede ser conveniente suponer que al expresar las palabras por polinomios, es numerar los dígitos de una palabra de longitud n de 0 a $n - 1$, en vez de 1 a n. La palabra $a_0a_1a_2a_3$ de longitud 4 es representada por el polinomio $a_0 + a_1x + a_2x^2 + a_3x^3$ de grado 3, por ejemplo.

Ejemplo 1.3. En la columna izquierda del esquema en la formación, el código $\mathcal{C}$ es representado por el polinomio en la columna de la derecha

Palabra código $\mathcal{C}$	Polinomio $c(x)$
0000	0
1010	$1 + x^2$
0101	$x + x^3$
1111	$1 + x + x^2 + x^3$

Podemos decir que $f(x)$ módulo $h(x)$ es $r(x)$, si $r(x)$ es el residuo cuando $f(x)$ es dividido por $h(x)$; podemos escribir $r(x) = f(x)$ mód $h(x)$. Además, decimos que dos polinomios $f(x)$ y $p(x)$ son equivalentes módulo $h(x)$ si y solamente si, ambos tienen el mismo residuo al ser divididos por $h(x)$; esto es si

$$f(x) \text{ mód } h(x) = r(x) = p(x) \text{ mód } h(x)$$

Denotamos esto por

$$f(x) \equiv p(x) \text{ mód } h(x)$$

Ejemplo 1.4. Sea $h(x) = 1 + x^5$ y $f(x) = 1 + x^4 + x^9 + x^{11}$. Entonces dividiendo $f(x)$ por $h(x)$ da por residuo $r(x) = 1 + x$. Podemos decir que $r(x) = f(x)$ (mód $h(x)$).

Similarmente, si $p(x) = 1 + x^6$, entonces $1 + x = 1 + x^6$ mód $(1 + x^5)$ y así podemos decir que $p(x) \equiv f(x)$ mód $h(x)$.

Ejemplo 1.5. Sea $p(x) = 1 + x + x^5$. Calculando $m(x)$ mód $p(x)$, con $m(x) = 1 + x^2 + x^6 + x^9 + x^{11}$ podemos encontrar el residuo $r(x) = x + x^3 + x^4$ y de ahí $x + x^3 + x^4 = m(x)$ mód $p(x)$. Note que si $m'(x) = x^2 + x^8$ entonces $m'(x)$ mód $p(x) = x^2 + x^3 + x^4$, donde $m'(x)$ y $m(x)$ no son equivalentes mód $p(x)$.

La adición y multiplicación de polinomios son compatibles "respecto" a la equivalencia de polinomios antes definido. Es decir:

Lema 1.1. Si $f(x) \equiv g(x)$ mód $h(x)$, entonces:

$$f(x) + p(x) \equiv (g(x) + p(x)) \text{ mód } h(x)$$
$$f(x).p(x) \equiv g(x).p(x) \text{ mód } h(x)$$

Demostración. Supongamos que $r(x) = f(x)$ mód $h(x)$ y $r(x) = g(x)$ mód $h(x)$ y $s(x) = p(x)$ mód $h(x)$ entonces por el algoritmo de la división tenemos

$$f(x) + p(x) = q_1(x)h(x) + r(x) + q_2(x)h(x) + s(x)$$
$$= (q_1(x) + q_2(x))\, h(x) + r(x) + s(x)$$
$$(f(x) + p(x)) \bmod h(x) = r(x) + s(x)$$
$$(f(x) + p(x)) \bmod h(x) = (g(x) + p(x)) \text{ (mód } h(x)) = r(x) + s(x),$$
puesto que grad$(r(x) + s(x)) <$ grad $h(x)$,

entonces concluimos: $f(x) + p(x) \equiv (g(x) + p(x))$ mód $h(x)$

Utilizaremos argumentos similares para el siguiente caso.

$$f(x).\, p(x) = (q_1(x)h(x) + r(x)).(q_2(x)h(x) + s(x))$$
$$= (q_1(x)s(x) + q_2(x)r(x)).h(x) + r(x).s(x)$$
$$(f(x).p(x)) \bmod h(x) = r(x).s(x)$$
$$(f(x).\, p(x)) \bmod h(x) = (g(x).p(x)) \text{ (mód } h(x)) = r(x).\, s(x)$$
puesto que grad$(r(x).s(x)) <$ grad $h(x)$

entonces concluimos: $f(x).p(x) \equiv g(x).p(x)$ mód $h(x)$ □

Ejemplo 1.6. Sea $h(x) = 1 + x^5$, $f(x) = 1 + x + x^7$, $g(x) = 1 + x + x^2$ y $p(x) = 1 + x^6$; así $f(x) \equiv g(x)$ (mód $h(x)$). Entonces

$$f(x) + p(x) = x + x^6 + x^7 \quad \text{y} \quad g(x) + p(x) = x + x^2 + x^6$$

pero

$$(x + x^6 + x^7) \text{ mód } h(x) = x^2 = (x + x^2 + x^6) \text{ mód } h(x)$$

Similarmente

$$(1 + x + x^7)(1 + x^6) \text{ mód } h(x) = 1 + x^3 = (1 + x + x^2)(1 + x^6) \text{ mód } h(x)$$

Note que $1 + x = (1 + x^6)$ mód $h(x)$. Así tenemos

$$(1 + x + x^7)(1 + x^6) \equiv (1 + x + x^2)(1 + x^6)$$
$$\equiv (1 + x + x^2)(1 + x)$$
$$\equiv 1 + x^3 \text{ (mód } h(x)).$$

1.3 Introducción a los Códigos Cíclicos

Pasamos ahora a estudiar una clase de códigos llamados códigos cíclicos. Después de todo, podemos usar lo que sabemos acerca de los códigos cíclicos para crear una matriz generadora para el código de corrección de errores BCH, al igual que para otros códigos. De hecho, podemos ver que tanto los códigos cíclicos como los códigos de Hamming son códigos cíclicos o similares a los códigos cíclicos.

Definición 1.6. Sea u una palabra de longitud n. El $Cambio\ cíclico\ \delta(u)$ de u, es la palabra de longitud n obtenido de u, toma el último dígito de u, lo traslada al principo y mueve el resto de los dígitos un lugar a la derecha. Por ejemplo

u	10110	111000	0000	1011
$\delta(u)$	01011	011100	0000	1101

Definición 1.7. Un código $\mathcal{C}$ es llamado a ser un *código cíclico,* si el cambio cíclico de cada palabra código es también una palabra código.

Ejemplo 1.7. El código $\mathcal{C} = \{000, 011, 101, 110\}$ es una código lineal cíclico. Primero $\mathcal{C}$ es lineal. Después calculamos $\delta(u)$ para todo u en $\mathcal{C}$.

$$\delta(000) = 000,\ \delta(011) = 101,\ \delta(101) = 110,\ \delta(110) = 011$$

puesto que $\delta(u)$ también está en $\mathcal{C}$, para cada u en $\mathcal{C}$, $\mathcal{C}$ es cíclico.

Ejemplo 1.8. El código $\mathcal{C} = \{000, 011, 100, 111\}$ no es cíclico. El cambio cíclico de $u = 011$ es $\delta(011) = 101$ el cuál no está en $\mathcal{C}$.

Note que el cambio cíclico δ es una transformación lineal; esto es,

Lema 1.2. $\delta(u + v) = \delta(u) + \delta(v)$ y $\delta(av) = a\delta(u)$, $a \in \mathcal{K} = \{0, 1\}$.

Para demostrar si un código lineal $\mathcal{C}$ es cíclico es suficiente demostrar si $\delta(u) \in \mathcal{C}$, en una base para $\mathcal{C}$ para cada palabra u.

Demostración. Sea $u = (u_0u_1 \dots u_{n-1})$, $v = (v_0v_1 \dots v_{n-1})$ entonces

$$u + v = (u_0 + v_0, u_1 + v_1, \dots, u_{n-1} + v_{n-1},)$$
$$\delta(u + v) = (u_{n-1} + v_{n-1}, u_0 + v_0, \dots, u_{n-2} + v_{n-2}) = \delta(u) + \delta(v)$$

□

Recordemos que un conjunto B no vacío de vectores en un espacio vectorial es una base para $\mathcal{C}$, si cumplen las siguientes condiciones: B expande a $\mathcal{C}$ ($\langle B\rangle = \mathcal{C}$) y B es un conjunto linealmente independiente. Se puede observar que cualquier conjunto B linealmente independiente es para $\langle B\rangle$ automáticamente una base.

Ejemplo 1.9. En el ejemplo 1.7, $\mathcal{C}$ es un código lineal ya que {110, 101} es una base para $\mathcal{C}$ puesto que $\delta(110) = 011$ y $\delta(101) = 110$ están en $\mathcal{C}$.

Si queremos crear un código lineal, primero debemos elegir una palabra u, formar un conjunto $\mathcal{S}$ de u y todos estos cambios cíclicos, $\mathcal{S} = \{u, \delta(u), \delta^2(u), \ldots, \delta^{n-1}(u)\}$ y definir $\mathcal{C}$ como un subespacio lineal de $\mathcal{S}$, es decir $\mathcal{C} = \langle\mathcal{S}\rangle$. (Podemos usar la notación $\delta^2(u) = \delta(\delta(u))$, $\delta^3(u) = \delta(\delta(\delta(u)))$, etc). $\mathcal{C}$ debe ser cíclico por lema 1.2, desde que $\mathcal{S}$ contiene una base para $\mathcal{C}$.

Ejemplo 1.10. Sea $n = 7$ y $u = 1101000$. Entonces $\mathcal{S}$= {u, $\delta(u)$, $\delta^2(u)$, $\delta^3(u)$, $\delta^4(u)$, $\delta^5(u)$, $\delta^6(u)$} = {1101000, 0110100, 0011010, 0001101, 1000110, 0100011, 1010001} y $\langle\mathcal{S}\rangle = \mathcal{K}^7$. Note que si $v = a_0u + a_1\delta(u) + a_2\delta^2(u) + a_3\delta^3(u) + a_4\delta^4(u) + a_5\delta^5(u) + a_6\delta^6(u)$ entonces

$$\delta(v) = a_0\delta(u) + a_1\delta^2(u) + a_2\delta^3(u) + a_3\delta^4(u) + a_4\delta^5(u) + a_5\delta^6(u) + a_6\delta^7(u)$$
$$= a_6u + a_0\delta(u) + a_1\delta^2(u) + a_2\delta^3(u) + a_3\delta^4(u) + a_4\delta^5(u) + a_5\delta^6(u)$$

Ejemplo 1.11. Sea $n = 6$ y $u = 010101$. Entonces $\delta(u) = 101010$ y $\delta^2(u) = 010101 = u$, así $\mathcal{S} = \{010101, 101010\}$ y $\mathcal{C} = \langle\mathcal{S}\rangle$ es el código cíclico, $\mathcal{C} = \{000000, 000111, 010101, 101010, 111000, 111111\}$.

Si una palabra u y su cambio cíclico forman un conjunto $\mathcal{S} = \{u, \delta(u), \ldots, \delta^{n-1}(u)\}$ el cual es un subespacio del código $\mathcal{C}$ ($\mathcal{C} = \langle\mathcal{S}\rangle$), entonces se puede decir que u es un *generador* de un código lineal cíclico $\mathcal{C}$. Podemos decir que $\mathcal{C}$ es el código cíclico más pequeño que contiene a u, dado que todo código lineal que contiene a u debe incluir a $\mathcal{S}$. Tenga en cuenta que un código cíclico lineal puede tener multiples generadores, debido a que un espacio vectorial usualmente tiene muchas bases.

Los códigos cíclicos tienen una forma muy conveniente de expresarlos en forma polinomial. Esto se basa en una simple observación que si una palabra u corresponde al polinomio $u(x)$ entonces el cambio cíclico de u, $\delta(u)$ le corresponde al polinomio $xu(x)$ mód$(1 + x^n)$. Tenga en cuenta que, en general $1 \equiv x^n$ (mód $(1 + x^n)$).

Ejemplo 1.12. Sea $u = 100$ entonces $u(x) = 1$ y $\delta(u) = 010$ corresponde a $xu(x) = x$. Similarmente si $u = 1101$ entonces $u(x) = 1 + x + x^3$ y $\delta(u) = 1110$ corresponde a $xu(x)$mód $(1 + x^4) = 1 + x + x^3$.

A los elementos de un código, para código cíclico nos referimos ambos como palabras código y polinomios. Ahora podemos repetir la discusión anterior de códigos cíclicos en forma polinomial. Se tiene una palabra u de longitud n, y se establece su correspondiente polinomio $u(x)$; por lo tanto los cambios cíclicos de u corresponden a los polinomios $x^i u(x)$ mód $(1 + x^n)$ para todo $i = 0, 1, \ldots, n-1$.

Ejemplo 1.13. Sea u = 1101000 y n = 7. Entonces $u(x) = 1 + x + x^3$ y

Palabra	Polinomio (mód $(1 + x^7)$)
0110100	$xu(x) = x + x^2 + x^4$
0011010	$x^2u(x) = x^2 + x^3 + x^5$
0001101	$x^3u(x) = x^3 + x^4 + x^6$
1000110	$x^4u(x) = x^4 + x^5 + x^7 \equiv 1 + x^4 + x^5$ (mód $(1 + x^7)$)
0100011	$x^5u(x) = x^5 + x^6 + x^8 \equiv x + x^5 + x^6$ (mód $(1 + x^7)$)
1000110	$x^6u(x) = x^6 + x^7 + x^9 \equiv 1 + x^2 + x^6$ (mód $(1 + x^7)$)

Claramente si $c(x) \in \langle\{u(x), xu(x), \ldots, x^{n-1}u(x)\}\rangle$, (mód $1 + x^n$) entonces eso significa que

$$\begin{aligned} c(x) &= (a_0u(x) + a_1xu(x) + \cdots + a_{n-1}x^{n-1}u(x)) \text{ mód } (1 + x^n) \\ &= (a_0 + a_1x + a_2x^2 + \cdots + a_{n-1}x^{n-1})u(x) \text{ mód } (1 + x^n) \\ &= a(x)u(x) \text{ mód } (1 + x^n) \end{aligned}$$

Entonces obtenemos el siguiente resultado:

Lema 1.3. Sea $\mathcal{C}$ un código cíclico y $u \in \mathcal{C}$, entonces cualquier polinomio $a(x)$, $c(x) = a(x)u(x)$ mód $(1 + x^n)$ es una palabra código en $\mathcal{C}$.

En toda palabra código distinta de cero de un código lineal $\mathcal{C}$, siempre va existir una palabra única $g' \in \mathcal{C}$, de modo que $g'(x)$ tiene un grado mínimo, definido por los siguientes argumentos.

$\mathcal{C}$ debe por lo menos de tener una palabra código o un polinomio de grado minimo. Suponiendo que hubiera otra palabra distinta de cero: g' y g'', corresponderían polinomios $g'(x)$ y $g''(x)$ con grado mínimo k, por lo tanto $g'(x) + g''(x) = c(x) \in \mathcal{C}$, ya que $\mathcal{C}$ es lineal y de grad$(c(x)) < k$ (ya que $x^k + x^k = 0$). Se sabe que g' es una palabra que tiene grado mínimo distinto de cero, grad$(c(x)) < k$, lo que significa que $c(x) = 0$, así $g'(x) = g''(x)$ y entonces $g'(x)$ es único.

En un código cíclico $\mathcal{C}$, se define el *polinomio generador* como el único polinomio mínimo distinto de cero. Sabemos que es único por la discusión anterior, pero no si es un generador.

Para ver esto, necesitamos evidenciar que para cualquier palabra código $c(x) \in \mathcal{C}$, existe

$a(x)$ tal que, $c(x) = a(x)g'(x)$ mód $(1 + x^n)$; de hecho, se demostrará que $c(x) = a(x)g'(x)$. Como el grad($c(x)$) ≥ grad($g'(x)$), podemos usar el algoritmo de la división

$$c(x) = q(x)g'(x) + r(x)$$

o

$$r(x) = q(x)g'(x) + c(x)$$

sin embargo ambos $c(x)$ y $q(x)g'(x)$ son palabras código de $\mathcal{C}$ por el Lema 1.3 y así es $r(x)$. Pero por el algoritmo de la división cualquiera $r(x) = 0$ o grad($r(x)$) < grad($g'(x)$). Puesto que lo último es imposible a no ser que $r(x) = 0$, concluimos que $r(x) = 0$ y así $g'(x)$ es un divisor de cada palabra código $c(x)$ en $\mathcal{C}$.

Teorema 1.1. *Sea $\mathcal{C}$ un código cíclico de longitud n y sea $g'(x)$ el polinomio generador. Si $n - k$ =* grad($g'(x)$), *entonces*

1. *$\mathcal{C}$ tiene dimensión k*
2. *La palabras código correspondientes a $g'(x)$, $xg'(x)$, . . . , $x^{k-1}g'(x)$ son una base para $\mathcal{C}$,* y
3. *$c(x) \in \mathcal{C}$ si y solamente si $c(x) = a(x)g'(x)$ para algún polinomio $a(x)$ con* grad($a(x)$) < k *(esto es, $g'(x)$ es un divisor de cada palabra código $c(x)$).*

Demostración. La discusión arriba del Teorema 1.1 prueba (3). Si $g'(x)$ tiene grado $n - k$ entonces $g'(x)$, $xg'(x)$, . . . , $x^{k-1}g'(x)$ debe ser linealmente independiente. Puesto que $g'(x)$ divide cada palabra código existe un único polinomio $a(x)=a_0+a_1x+\cdots+a_{k-1}x^{k-1}$ tal que $c(x) = a(x)g'(x) = a_0g'(x) + a_1xg'(x) + \cdots + a_{k-1}x^{k-1}g'(x)$. Por lo tanto $c(x)$ esta en $\langle\{g'(x), xg'(x), \ldots, x^{k-1}g'(x)\}\rangle$ y así $\{g'(x), xg'(x), \ldots, x^{k-1}g'(x)\}$ es una base para $\mathcal{C}$. □

Ejemplo 1.14. Sea $n = 7$, $g'(x) = 1 + x + x^3$ el generador para el código cíclico $\mathcal{C}$. Una base para $\mathcal{C}$ es:

$$\begin{aligned} g'(x) &= 1 + x + x^3 &\leftrightarrow\ & 1101000 \\ xg'(x) &= x + x^2 + x^4 &\leftrightarrow\ & 0110100 \\ x^2g'(x) &= x^2 + x^3 + x^5 &\leftrightarrow\ & 0011010 \\ x^3g'(x) &= x^3 + x^4 + x^6 &\leftrightarrow\ & 0001101 \end{aligned}$$

Note que $x^4g'(x)$ mód $1+x^7 = 1+x^4+x^5$ es una palabra código puesto que $1+x^4+x^5 = (1 + x + x^2)(1 + x + x^3) = (1 + x + x^2)g'(x)$.

Ejemplo 1.15. Sea $\mathcal{C}$ el código cíclico $\mathcal{C} = \{0000, 1010, 0101, 1111\}$; los polinomios correspondientes son $\{0, 1 + x^2, x + x^3, 1 + x + x^2 + x^3\}$. Note que, $1 + x^2 \leftrightarrow 1010$ es el polinomio generador para $\mathcal{C}$, puesto que $\mathcal{C}$ contiene solamente un polinomio de grado 2 y ninguno de grado 1. Además, cada palabra (polinomio) en $\mathcal{C}$ es un múltiplo del polinomio generador:

$$0 = 0(1 + x^2) \qquad x + x^3 = x(1 + x^2)$$
$$1 + x^2 = 1(1 + x^2) \qquad 1 + x + x^2 + x^3 = (1 + x)(1 + x^2)$$

Ejemplo 1.16. El código lineal cíclico más pequeño de longitud 6 conteniendo a $g'(x) = 1 + x^3 \leftrightarrow 100100$ es

$$\{000000, 100100, 010010, 001001, 110110, 101101, 011011, 111111\}$$

Esto se puede verificar por las técnicas descritas en esta sección. El polinomio de grado mínimo que representa una palabra en $\mathcal{C}$ se puede ver observando $g'(x) = 1 + x^3$, y que $\mathcal{C}$ no tiene ningún otro polinomio de grado 3. Así $g'(x) = 1 + x^3$ es el polinomio generador para $\mathcal{C}$. Representamos cada palabra en $\mathcal{C}$ como un múltiplo de $g'(x)$,

Palabra	Polinomio $f(x)$	factorización $h(x)g'(x)$ de $f(x)$
000000	0	$0(1 + x^3)$
100100	$1 + x^3$	$1(1 + x^3)$
010010	$x + x^4$	$x(1 + x^3)$
001001	$x^2 + x^5$	$x^2(1 + x^3)$
110110	$1 + x + x^3 + x^4$	$(1 + x)(1 + x^3)$
101101	$1 + x^2 + x^3 + x^5$	$(1 + x^2)(1 + x^3)$
011011	$x + x^2 + x^4 + x^5$	$(x + x^2)(1 + x^3)$
111111	$1 + x + x^2 + x^3 + x^4 + x^5$	$(1 + x + x^2)(1 + x^3)$

Podemos generar fácilmente códigos cíclicos simplemente eligiendo una palabra u y configurando $\mathcal{C} = \langle\{u(x), xu(x), \ldots, x^{n-1}u(x)\}\rangle$ (mód $1 + x^n$). Sin embargo, necesitamos encontrar el polinomio generador para dicho código y enumerar todas las palabras código no sería un enfoque prudencial. Para un código cíclico el polinomio generador tiene una importante propiedad:

Teorema 1.2. *$g'(x)$ es el polinomio generador para un código lineal cíclico de longitud n, si y sólo si $g'(x)$ divide $1 + x^n$ (así $1 + x^n = h(x)g'(x)$).*

Demostración. Por el algoritmo de la división $1 + x^n = h(x)g'(x) + r(x)$ con $r(x) = 0$

o grad($r(x)$) < grad($g'(x)$). Equivalentemente $r(x) = h(x)g'(x) + (1 + x^n)$. Pero $r(x) = (h(x)g'(x) + (1 + x^n))$ mód $(1 + x^n) = h(x)g'(x)$ mód $(1 + x^n)$. Así $r(x)$ está en el código generado por $g'(x)$ y $r(x) = 0$ o grad($r(x)$) ≤ grad($g'(x)$). Podemos concluir que $r(x) = 0$. □

Corolario 1.2.1. El polinomio generador $g'(x)$ para el código cíclico mínimo de longitud n conteniendo la palabra u (polinomio $u(x)$), es el máximo común divisor de $u(x)$ y $1 + x^n$ (es decir, $g'(x)$ = mcd($u(x)$, $1 + x^n$)).

Demostración. Si $g'(x)$ es un polinomio generador entonces $g'(x)$ divide a $u(x)$ y $1 + x^n$. Pero $g'(x) \in \langle\{u(x), xu(x), \ldots, x^{n-1}u(x)\}\rangle$, así tenemos

$$g'(x) = a(x)u(x) \text{ (mód } 1 + x^n)$$

o equivalentemente, por el Algoritmo de la División

$$g'(x) = a(x)u(x) + b(x)(1 + x^n).$$

Por lo tanto cualquier divisor común de $u(x)$ y $1 + x^n$ debe dividir $g'(x)$ y de esta manera $g'(x)$ es el máximo común divisor. □

Ejemplo 1.17. Sea $n = 8$ y $u = 11011000$, es decir $u(x) = 1 + x + x^3 + x^4$. El mcd de $u(x)$ y $1 + x^8$ es $1 + x^2$. Así $g'(x) = 1 + x^2$ y el menor código lineal cíclico conteniendo $u(x)$ tiene dimensión de 6 y $g'(x)$ como el polinomio generador.

Encontrar una alternativa razonable al polinomio generador para un código cíclico de longitud n y dimensión $n - k$ implica una simple reducción de filas. Tomando una matriz generadora (o base) y se coloca en una *forma escalonada reducida* (FER) con las últimas k columnas como "Columnas líderes", entonces la fila de grado mínimo (palabra código) se convierte en el polinomio generador.

1.4 Codificación y decodificación de polinomios

Para códigos cíclicos lineales se pueden encontrar diversas matrices generadoras; las más simple son las matrices donde las palabras código correspondientes al polinomio generador son las filas y sus $k - 1$ cambios cíclicos:

$$G' = \begin{bmatrix} g'(x) \\ xg'(x) \\ \vdots \\ x^{k-1}g'(x) \end{bmatrix}$$

Ejemplo 1.18. Sea un código lineal cíclico $\mathcal{C} = \{0000, 1010, 0101, 1111\}$, con $g'(x) = 1 + x^2$, como polinomio generador. Si $n = 4$ y $k = 2$, entonces una base para $\mathcal{C}$ sería

$$g'(x) = 1 + x^2 \leftrightarrow 1010, \quad xg'(x) = x + x^3 \leftrightarrow 0101$$

la cual puede ser verificado sencillamente. La matriz generadora para $\mathcal{C}$ sería

$$G' = \begin{bmatrix} g'(x) \\ xg'(x) \end{bmatrix} = \begin{bmatrix} 1010 \\ 0101 \end{bmatrix}$$

Ejemplo 1.19. Sea un código cíclico lineal $\mathcal{C}$ cuya longitud $n = 7$ y $g'(x) = 1 + x^2 + x^3$ polinomio generador, $\text{grad}(g'(x)) = n - k = 3$, por lo tanto $k = 4$, así una base para $\mathcal{C}$ es,

$$g'(x) = 1 + x^2 + x^3$$
$$xg'(x) = x + x^3 + x^4$$
$$x^2g'(x) = x^2 + x^4 + x^5$$
$$x^3g'(x) = x^3 + x^5 + x^6$$

y, su matriz generadora para $\mathcal{C}$ es

$$G' = \begin{bmatrix} 1011000 \\ 0101100 \\ 0010110 \\ 0001011 \end{bmatrix}$$

Dado que $\mathcal{C}$ es un código lineal de longitud n y k - dimensión (por lo que el polinomio generador $g'(x)$ tiene grado $n - k$). Los k dígitos de información $(a_0, a_1, \ldots, a_{k-1})$ a codificar puede considerarse como un polinomio $a(x) = a_0 + a_1x + \cdots + a_{k-1}x^{k-1}$ la cual se denomina *polinomio de información o mensaje*. La codificación consiste únicamente en la multiplicación de polinomios; es decir, $a(x)$ se codifica como $a(x)g'(x) = c(x)$. Por lo tanto, en lugar de almacenar $k \times n$ entradas en la matriz generadora, solo se almacena el polinomio generador, lo que es una mejora significativa en la complejidad de la codificación.

El inverso de la multiplicación de polinomios es la división de polinomios. A partir de aquí encontramos el correspondiente mensaje a la palabra código $c(x)$ más cercana al mensaje recibido, que consiste en $c(x)$ dividido por $g'(x)$, por lo que recuperamos el *polinomio mensaje* $a(x)$.

Ejemplo 1.20. Sea $g'(x) = 1 + x^2 + x^3$ y $n = 7$. Entonces $k = 7-3 = 4$. Sea $a(x) = 1 + x^3$ el polinomio mensaje que le corresponde a la palabra $a = 1001$. El mensaje $a(x)$ es codificado como $c(x) = a(x)g'(x)$, así

$$c(x) = (1 + x^3)(1 + x^2 + x^3) = 1 + x^2 + x^5 + x^6$$

con $c = 1010011$ como la palabra código correspondiente.

Si $c(x) = 1 + x + x^2 + x^4$ entonces el polinomio mensaje es $c(x)/g'(x) = a(x) = 1 + x$ correspondiente al mensaje $a = 1100$.

Habiendo desarrollado un procedimiento de codificación polinómica para códigos lineales cíclicos debemos considerar una matriz de verificación de paridad para tales códigos como un algoritmo para decodificar palabras recibidas. Si $c(x)$ es enviado y $w(x)$ es recibido, con $w(x) = c(x) + e(x)$ entonces uno gustaría calcular el síndrome y el polinomio error más probable $e(x)$.

El *síndrome polinómico*, $s(x)$, es definido por $s(x) = w(x)$ mód $g'(x)$. Asumiendo $g'(x)$ de grado $n - k$, entonces $s(x)$ tiene grado menor que $n - k$ y corresponderá a la palabra binaria s, de longitud $n - k$. Puesto que $w(x) = c(x) + e(x)$ y $c(x) = a(x)g'(x)$ tenemos que $s(x) = e(x)$ mód $g'(x)$. Esto es, el síndrome polinómico es independiente únicamente del error.

Podemos definir una matriz H en la cual la $i-$ésima fila r_i es la palabra de longitud $n - k$ correspondiente a $r_i \equiv x^i$ mód $g'(x)$. Resulta que esta matriz en una matriz de verificación de paridad para el código. Pero, si w es una palabra recibida entonces

$$w(x) \equiv c(x) + e(x)$$

$$wH = (c + e)H$$

$$wH = \sum_{i=0}^{n-1} (c_i + e_i) r_i$$

$$\leftrightarrow \sum_{i=0}^{n-1} (c_i + e_i) r_i(x)$$

$$wH = \left(\sum_{i=0}^{n-1} c_i x^i \right) \ m\acute{o}d\ g'(x) + \left(\sum_{i=0}^{n-1} e_i x^i \right) \ m\acute{o}d\ g'(x)$$

$$wH = c(x) \ m\acute{o}d\ g'(x) + e(x) \ m\acute{o}d\ g'(x)$$

$$wH = 0 + e(x) \ m\acute{o}d\ g'(x)$$

$$wH = s(x).$$

Entonces $s(x) = 0$ si y solo si $w(x)$ es una palabra código, así H es una matriz de verificación de paridad. Además, si $wH = s$ entonces s corresponde a $s(x) = w(x)$ mód $g'(x)$. Es claro que nos referimos a $s(x)$ como el síndrome polinómico.

Ejemplo 1.21. Sea $n = 7$, y $g'(x) = 1 + x + x^3$. Entonces $n - k = 3$. Producimos H como sigue:

$$\begin{aligned} r_0(x) &= 1 \quad \text{mód } g'(x) = 1 && \leftrightarrow 100 \\ r_1(x) &= x \quad \text{mód } g'(x) = x && \leftrightarrow 010 \\ r_2(x) &= x^2 \quad \text{mód } g'(x) = x^2 && \leftrightarrow 001 \\ r_3(x) &= x^3 \quad \text{mód } g'(x) = 1 + x && \leftrightarrow 110 \\ r_4(x) &= x^4 \quad \text{mód } g'(x) = x + x^2 && \leftrightarrow 011 \\ r_5(x) &= x^5 \quad \text{mód } g'(x) = 1 + x + x^2 && \leftrightarrow 111 \\ r_6(x) &= x^6 \quad \text{mód } g'(x) = 1 + x^2 && \leftrightarrow 101 \end{aligned}$$

así

$$H = \begin{bmatrix} 100 \\ 010 \\ 001 \\ 110 \\ 011 \\ 111 \\ 101 \end{bmatrix}$$

Si $w(x) = 1 + x^5 + x^6$ es recibida, $w = 1000011$, entonces $wH = s = 110$ y $s(x) = 1 + x = 1 + x^5 + x^6$ mód $(1 + x + x^3)$.

En vez de construir una matriz de decodificación estándar (SDA) para un código cíclico usaremos un algoritmo que utiliza simetrías inherentes en códigos cíclicos. Note que si e es un cociente líder y si $s = eH$, entonces $s(x) = e(x)$ mód $g'(x)$ como mostramos en el Ejemplo 1.21. Entonces $x^i s(x) \equiv x^i e(x)$ mód $g'(x)$ y así los síndromes de las partes cíclicas de e son fácilmente calculadas; en vez de almacenar un SDA usaremos esta propiedad.

Es importante notar que si grad $e(x) <$ grad $g'(x)$ entonces $e(x) = e(x)$ mód $g'(x)$ y así el síndrome polinómico para el polinomio error $e(x)$ es solo $e(x)$ (es decir, $s(x) = e(x)$). También podemos notar que si $e(x)$ es un cociente líder para un código cíclico de longitud n, así $x^i e(x)$ mód $(1 + x^n)$.

Algoritmo 1.1. (Para decodificar códigos cíclicos)

1. Calcular el síndrome polinómico $s(x) = w(x)$ mód $g'(x)$, donde w es la palabra recibida.

2. Para cada $i \geq 0$, calcular $s_i \leftrightarrow s_i(x) = x^i s(x)$ mód $g'(x)$ (el síndrome polinómico del $i-$ésimo parte cíclica de w) hasta el síndrome s_j es encontrado con peso $(s_j) \leq t$. Entonces el polinomio error más probable es $e(x) = x^{n-j} s_j(x)$ mód $(1 + x^n)$.

Observación 1.1. Este algoritmo de decodificación solo será correcto para patrones de error $e(x)$ donde, para algún i, $x^i e(x)$ mód $(1 + x^n)$ tiene grado a la sumo $n - k$. Esto es posible puesto que existen patrones de error de peso a lo sumo t que no satisfacen esta propiedad. Tales patrones de error son corregibles por el código, pero las palabras códigos más cercanas no se pueden encontrar con este algoritmo.

Ejemplo 1.22. Sea $n = 7$, sea $g'(x) = 1 + x + x^3$ el polinomio generador para el 1-código corrector de error lineal cíclico (así $t = 1$). Si $w(x) = x^2 + x^3$ es recibido entonces $s(x) = w(x)$ mód $g'(x) = x^2 + x^3$ mód $(1 + x + x^3) = 1 + x + x^2$ es el síndrome polinómico. Calculamos

$$s_1(x) = xs(x) \text{ mód } g'(x) = x(1 + x + x^2) \text{ mód } g'(x) = 1 + x^2$$
$$s_2(x) = x^2 s(x) \text{ mód } g'(x) = x(1 + x^2) \text{ mód } g'(x) = 1,$$

que tienen peso $1 \leq t$. Así $j = 2$ y por lo tanto

$$e(x) = x^{7-2} s_2(x) \text{ mód } (1 + x^7) = x^5$$

Así $c(x) = w(x) + e(x) = (x^2 + x^3) + x^5$ es la palabra código más probable.

Ejemplo 1.23. Sea $n = 15$, y sea $g'(x) = 1 + x^4 + x^6 + x^7 + x^8$ el polinomio generador para el código cíclico con $d = 5$. Así todos los patrones de error con peso $t = 2$ o menos son corregibles. Decodificar la palabra recibida $w = 110011100111000$.
Aquí $w(x) = 1 + x + x^4 + x^5 + x^6 + x^9 + x^{10} + x^{11}$.
El síndrome polinómico $s(x) = w(x)$ mód $g'(x)$ es

$$s(x) = 1 + x + x^3 + x^4 + x^5 + x^6 + x^7$$
$$s_1(x) = xs(x) = x + x^2 + x^4 + x^5 + x^6 + x^7 + x^8 \text{ mód } g'(x)$$
$$= 1 + x + x^2 + x^5$$
$$s_2(x) = x^2 s(x) \equiv x + x^2 + x^3 + x^6 \text{ (mód } g'(x))$$
$$s_3(x) = x^3 s(x) \equiv x^2 + x^3 + x^4 + x^7 \text{ (mód } g'(x))$$
$$s_4(x) = x^4 s(x) \equiv 1 + x^3 + x^5 + x^6 + x^7 \text{(mód } g'(x))$$
$$s_5(x) = x^5 s(x) \equiv 1 + x \text{ (mód } g'(x))$$

los cuales tienen peso $2 \leq t$.
Así $e(x) = x^{15-5}s_5(x)$ mód $(1 + x^{15}) = x^{10} + x^{11}$.
Por lo tanto

$$\begin{aligned} c(x) &= w(x) + e(x) \\ &= w(x) + (x^{10} + x^{11}) \\ &= 1 + x + x^4 + x^5 + x^6 + x^9 \end{aligned}$$

1.5 Encontrando códigos cíclicos

Si se quiere construir un código lineal de longitud n y dimensión k, se debe encontrar un factor de $1 + x^n$ que tiene grado $n - k$. Por supuesto que puede haber varios casos o ninguno para n y k dados. Existe la pregunta de la distancia mínima para códigos cíclicos que no hemos considerado, una pregunta que no está resuelta en general. La postergaremos para después.
Para reiterar, el hecho que cada generador debe dividir $1 + x^n$ nos permite encontrar todos los códigos lineales cíclicos de una longitud *n* dada. Todo lo que tenemos que hacer es encontrar todos los factores de $1 + x^n$, que significa que primero debemos encontrar todos los factores irreducibles.
Sea el polinomio $f(x) \in \mathcal{K}[x]$ de grado 1 como mínimo, además no es el producto de dos polinomios $\in \mathcal{K}[x]$, ambos de grado 1 como mínimo, entonces $f(x)$ es irreducible. Encontrar todos los factores irreducibles (casi todos los factores que dan son de $1 + x^n$) no es tan sencillo. La potencia de 1, como factor de $1 + x^n$ es 0, por lo que se genera un código cíclico de n - dimensión; este código debe ser $\mathcal{K}^n$, lo que prueba que $\mathcal{K}^n$ es cíclico. También podemos definir el código $\{0\}$ que consta unicamente de la palabra cero de longitud n de una manera especial para ser cíclico con generador $g'(x) = 0 = 1 + x^n$ (mód $1 + x^n$).
Podemos llamar a estos códigos lineales cíclicos $\mathcal{K}^n$ y $\{0\}$, *códigos cíclicos impropios*. De otra manera, el código es un *código cíclico propio*.

Ejemplo 1.24. Para $n = 3$, se tiene que $1 + x^3 = (1 + x)(1 + x + x^2)$ es la factorización de $1 + x^3$ en factores irreducibles. Por lo tanto hay dos códigos cíclicos propios de longitud 3. Uno tiene como generador:

$g'(x) = 1 + x$ y matriz que lo genera

$$G' = \begin{bmatrix} 110 \\ 011 \end{bmatrix}$$

El código es $\mathcal{C} = \{000, 110, 011, 101\}$. El otro código tiene generador $g'(x) = 1 + x + x^2$ y matriz generadora $G' = [111]$, así es el código $\mathcal{C} = \{000, 111\}$

Ejemplo 1.25. Para $n = 6$, la factorización de $1 + x^6$ en sus factores irreducibles es:

$$1 + x^6 = (1 + x^3)^2 = (1 + x)^2(1 + x + x^2)^2.$$

Por lo tanto, para encontrar los generadores de códigos lineales cíclicos de longitud 6, establecemos todos los posibles productos de estos factores exceptuado para 1 y $1 + x^6$. Cada producto es el generador para un código lineal cíclico de longitud 6. Las dimensiones y la longitud de estos productos del código lineal cíclico de longitud 6 se muestran en la tabla para cada producto resultante

generador	dimensión
$1 + x$	5
$(1 + x)^2 = 1 + x^2$	4
$1 + x + x^2$	4
$(1 + x + x^2)^2 = 1 + x^2 + x^4$	2
$(1 + x)(1 + x + x^2) = 1 + x^3$	3
$(1 + x)^2(1 + x + x^2) = 1 + x + x^3 + x^4$	2
$(1 + x)(1 + x + x^2)^2 = 1 + x + x^2 + x^3 + x^4 + x^5$	1

Teorema 1.3. *Si* $n = 2^r s$ *entonces* $1 + x^n = (1 + x^s)^{2^r}$

Demostración. Si $n = 2s$, entonces $(1 + x^s)^2 = 1 + x^s + x^s + x^{2s} = 1 + x^{2s}$. Entonces procedemos por inducción sobre r. □

Corolario 1.3.1. Sea $n = 2^r s$, donde s es par y sea $1 + x^s$ el producto de z polinomios irreducibles, entonces existen $(2^r + 1)^z$ códigos lineales cíclicos de longitud n y $(2^r + 1)^z - 2$ códigos lineales cíclicos propios de longitud n.

Ejemplo 1.26. En el Ejemplo 1.24 está demostrado que $1 + x^3$ es el producto de dos polinomios irreducibles, a saber $1 + x$ y $1 + x + x^2$. Aplicando el Corolario 1.17 con $r = 0$, $s = 3$ y $z = 2$ encontramos que existen $(2^0 + 1)^2 = 4$ códigos lineales cíclicos de longitud 3, 2 de los cuales son propios (como se muestra en el Ejemplo 1.24). Además, para $1 + x^6$, tenemos $n = 6 = 2^1.3$ así $r = 1$, z es aún 2 así existen $(2 + 1)^2 = 9$ códigos lineales cíclicos de longitud 6, 7 de los cuales son propios (como se muestra en el Ejemplo 1.25)

Uno puede encontrar códigos cíclicos o equivalentes al factor $1+x^n$, por un procedimiento relativamente simple. A través de nuestra discusión asumiremos que para todo n es par. Como primer paso se debe generar todos los polinomios $I(x)$ mód $(1 + x^n)$ tales que $I(x) = I(x)^2$ mód $(1 + x^n)$. Estos polinomios se llaman *polinomios idempotentes.* Fácilmente entendemos que si $u(x)$ y $v(x)$ son idempotentes, entonces su suma $u(x)$ + $v(x)$ y su producto $u(x)v(x)$ mód $(1 + x^n)$ también son idempotentes. Así necesitamos construir solamente un conjunto base de polinomios idempotentes. Para hacer esto necesitamos la partición de los $Z_n = \{0, 1, 2, \ldots, n - 2, n - 1\}$ en clases.
Sea $\mathcal{C}_i = \{s = 2^j \cdot i (\text{mód } n) / j = 0, 1, \ldots, r\}$, además 2^r mód $n = 1$.

Ejemplo 1.27. Para $n = 7$ tenemos

$$\mathcal{C}_0 = \{0\},\ \mathcal{C}_1 = \{1, 2, 4\} = \mathcal{C}_2 = \mathcal{C}_4,\ \text{y}\ \mathcal{C}_3 = \{3, 5, 6\} = \mathcal{C}_5 = \mathcal{C}_7.$$

Para $n = 9$ tenemos

$$\mathcal{C}_0 = \{0\},\ \mathcal{C}_1 = \{1, 2, 4, 8, 7, 5\},\ \text{y}\ \mathcal{C}_3 = \{3, 6\}$$

Siguiendo para cada clase diferente $\mathcal{C}_i$ formamos un polinomio

$$c_i(x) = \sum_{j \in \mathcal{C}_i} x^j$$

Afirmamos que $c_i(x)$ es idempotente y además que cualquier idempotente $I(x)$(mód $1+ x^n$) es

$$I(x) = \sum_{i=0}^{k} a_i c_i(x),\ a_i \in \{0,1\}$$

Para ver esto note que,

$$c_i(x)^2 = c_i(x^2) = \sum_{j \in \mathcal{C}_i} x^{2j} = \sum_{k \in \mathcal{C}_i} x^k (\ mód\ 1 + x^n)$$

puesto que si $j \in \mathcal{C}_i$ entonces también lo es $2j$(mód n).

Ejemplo 1.28. Para $n = 7$ tenemos,

$\mathcal{C}_0 = \{0\}$, entonces $c_0(x) = x^0 = 1$
$\mathcal{C}_1 = \{1, 2, 4\}$, entonces $c_1(x) = x^1 + x^2 + x^4$
$\mathcal{C}_3 = \{3, 5, 6\}$, entonces $c_3(x) = x^3 + x^6 + x^5$.

Entonces cualquier polinomio idempotente mód $(1 + x^7)$ puede ser expresado como

$$I(x) = a_0c_0(x) + a_1c_1(x) + a_3c_3(x),\ a_i \in \{0, 1\}.$$

Así tenemos $2^3 - 1$ idempotentes diferentes mód $(1 + x^n)$. (Ignoramos que $I(x) = 0$ que es el idempotente trivial).

La relación entre códigos idempotentes y cíclicos es:

Teorema 1.4. *Cada código cíclico contiene un polinomio idempotente único que genera el código.*

Demostración. Sea $g'(x)$ el generador de un código cíclico de longitud n y sea $g'(x)h(x) = 1 + x^n$ (n es par). Entonces $MCD(h(x), g'(x)) = 1$ y según el Algoritmo de Euclides existen polinomios $t(x)$, $s(x)$ tal que

$$1 = t(x)\, g'(x) + s(x)h(x)$$

Multiplicando ambos lados por $t(x)\, g'(x)$ resulta,

$$t(x)\, g'(x) = (t(x)\, g'(x))^2 + t(x)s(x)(1 + x^n)$$

o

$$t(x)\, g'(x) = (t(x)\, g'(x))^2 \text{ mód } 1 + x^n.$$

Así $t(x)\, g'(x)$ es un idempotente y

$$g'(x) = MCD(t(x)\, g'(x), 1 + x^n).$$

□

Capítulo 2

Códigos BCH

2.1 Utilidad en los Campos Finitos

Veremos en este capítulo códigos cíclicos de una clase especial y otra forma de decodificarlos utilizando el campo de Galois *GF* (2^r); que es un cuerpo con un número finito de elementos, llamado también cuerpo finito definido sobre un conjunto de elementos finito, la cual el número de elementos debe ser un número primo o potencia de un número primo.

Sabemos que el polinomio $d(x)$ es un divisor o factor de $f(x)$ si $f(x) = g'(x)d(x)$. Además, siempre los divisores triviales de $f(x)$ son 1 y $f(x)$. Se llama *divisor no trivial o propio* de $f(x)$ a cualquier otro divisor. Se dice que un polinomio $f(x) \in \mathcal{K}[x]$ es *irreducible* en $\mathcal{K}$ si no tiene divisores propios en $\mathcal{K}[x]$; de lo contrario, es reducible (o divisible) en $\mathcal{K}$.

Ejemplo 2.1. Por definición, los polinomios x y $1 + x$ son irreducibles; $1 + x + x^2$ no tiene a x ni a $1 + x$ como divisores, por lo que también es irreducible. Sin embargo, x^2, $1 + x^2$ y $x + x^2$ son no irreducibles: x^2 y $x + x^2$ tienen a *x* como divisor; $1 + x^2$ tiene a $1 + x$ como divisor.

En general, $1 + x$ es un divisor o factor de $f(x)$ si y solo si 1 es raíz de $f(x)$; lo que significa; si y solo si $f(1) = 0$. Tenga en cuenta que $1 + x$ es un factor de $f(x) = 1 + x^2$ y $f(1) = 1 + 1 = 0$. De manera similar, x es un factor de $g'(x)$ si y solo si $g'(0) = 0$. Pero encontrar otros factores irreducibles de polinomios es más difícil, por ahora es solo un asunto de prueba y error.

Ejemplo 2.2. Si $f(x) = 1 + x + x^2 + x^3$, entonces $f(1) = 1 + 1 + 1 + 1 = 0$ y así $1 + x$ es un factor de $f(x)$. Por división larga $f(x) = (1 + x)(1 + x^2) = (1 + x)^3$. Por otro lado,

si $g'(x) = 1 + x + x^3$, entonces $g'(0) = 1 \neq 0$ y $g'(1) = 1 \neq 0$, entonces $g'(x)$ no cuenta con factores lineales. Por lo tanto $g'(x)$ es irreducible sobre $\mathcal{K}$, ya que para que sea reducible un polinomio cúbico un factor lineal debe tener.

Ejemplo 2.3. Sea $f(x) = 1 + x + x^4$. Como $f(0) \neq 0$ y $f(1) \neq 0$, $f(x)$ no tiene factores lineales. Por lo tanto, si $f(x)$ es reducible, entonces $f(x)$ debe tener factores cuadráticos irreducibles. Sabemos que el único factor cuadrático irreducible en $\mathcal{K}$ es $g'(x) = 1 + x + x^2$. Luego de dividir $g'(x)$ por $f(x)$, encontraremos un residuo distinto de cero. Entonces $1 + x + x^2$ no es un factor de $f(x)$. Por lo tanto $f(x)$ es irreducible sobre $\mathcal{K}$.

Un polinomio irreducible de grado n, $n > 1$ sobre $\mathcal{K}$ se llama *polinomio primitivo* si no es divisor de $1 + x^m$ para cualquier $m < 2^n - 1$. Veremos que los polinomios irreducibles de grado *n* siempre divide a $1 + x^m$ cuando $m = 2^n - 1$.

Ejemplo 2.4. Ya que $1 + x + x^2$ no es un factor de $1 + x^m$ para $m < 3 = 2^2 - 1$, es primitivo. Además, para cualquier $m < 7 = 2^3 - 1$, $1 + x + x^3$ no es un factor de $1 + x^m$ y, por lo tanto, es primitivo.
Sin embargo $1 + x^5 = (1 + x)(1 + x + x^2 + x^3 + x^4)$ y $1 + x + x^2 + x^3 + x^4$ son irreducibles, pero $5 < 15 = 2^4 - 1$ y así $1 + x + x^2 + x^3 + x^4$ no es primitivo.

Recuerda que la adición y la multiplicación de polinomios módulos para un polinomio $h(x)$ de grado n si se puede definir. Tenemos $\mathcal{K}^n[x]$, que representa a todos los polinomios en conjunto de $\mathcal{K}$[x] de menor grado n. Claro está, que cada palabra en $\mathcal{K}^n$ le corresponde un polinomio en $\mathcal{K}^n[x]$ por lo que podemos definir la adición y multiplicación de palabras en $\mathcal{K}^n$. Podemos introducir en este capítulo la estructura adicional de campos finitos para ayudar en la decodificar códigos codificados.

Nosotros ya tenemos una definición de adición y multiplicación de palabras en $\mathcal{K}^n$, pero para que esto forme un campo necesitamos tener cuidado en nuestro caso de $h(x)$. Por ejemplo, en un campo debe ser el caso que si $ab = 0$ entonces $a = 0$ o $b = 0$.

Ejemplo 2.5. Podemos definir la multiplicación de palabras en $\mathcal{K}^4$ usando la multiplicación de polinomios módulo $1 + x^4$. Sin embargo,

$$\begin{aligned}(0101)(0101) &\leftrightarrow (x + x^3)(x + x^3)\\ &= x^2 + x^6\\ &= (x^2 + x^2)(\text{mód } 1 + x^4)\\ &= 0\\ &\leftrightarrow 0000,\end{aligned}$$

así (0101)(0101) = 0000, pero 0101 ≠ 0000 en $\mathcal{K}^4$. Entonces $\mathcal{K}^4$ no puede ser un campo de acuerdo con esta multiplicación definida.

La complicación en el ejemplo último surge debido a que $1 + x^4$ sobre $\mathcal{K}$ no es irreducible. Una manera de definir la multiplicación en $\mathcal{K}^n$ y hacer de $\mathcal{K}^n$ un campo es definir la multiplicación en $\mathcal{K}^n$ módulo como un polinomio irreducible de grado n. Dejamos la prueba que $GF(2^r)$ es un campo para un curso de algebra moderna.

Ejemplo 2.6. Usando el polinomio irreducible $h(x) = 1 + x + x^4$, definiendo la multiplicación en $\mathcal{K}^4$. Para encontrar el producto (1101)(0101) notemos que

$$(1101)(0101) \quad \leftrightarrow \quad (1 + x + x^3)(x + x^3)$$

Pero $(1 + x + x^3)(x + x^3) = x + x^2 + x^3 + x^6$ y $x = x + x^2 + x^3 + x^6$ mód $(1 + x + x^4)$. Así (1101)(0101) = 0100 ↔ x

Ejemplo 2.7. Usando el polinomio primitivo $h(x) = 1 + x + x^3$ para definir la multiplicación, considere construir $GF(2^3)$. Hacemos este cálculo x^i mód $h(x)$:

palabra ↔	x^i mód $h(x)$
100	1
010	x
001	x^2
110	$x^3 \equiv 1 + x$
011	$x^4 \equiv x + x^2$
111	$x^5 \equiv 1 + x + x^2$
101	$x^6 \equiv 1 + x^2$

Para calcular (110)(001) ↔ $(1 + x)x^2$ note que de la tabla anterior $1 + x = x^3$ mód $h(x)$ así

$$\begin{aligned}(x^2)(1 + x) &\equiv x^2 \cdot x^3 \\ &\equiv x^5 \\ &\equiv 1 + x + x^2 \ (\text{mód } h(x))\end{aligned}$$

luego

$$(110)(001) = 111$$

Para construir $GF(2^r)$ usando el polinomio primitivo hace los cálculos en el campo mucho más fáciles que usando un polinomio irreducible no primitivo. Para ver esto:

Sea $\beta \in GF(2^r)$ que representa la palabra correspondiente para x mód $h(x)$, donde $h(x)$ es un polinomio primitivo de grado n, entonces $\beta^i \leftrightarrow x^i$ mód $h(x)$. Tenga en cuenta que $1 = x^m$ mód $h(x)$ significa que $0 = 1 + x^m$ mód $h(x)$ y así que $h(x)$ divide $1 + x^m$. Como que $h(x)$ es primitivo, sabemos que $h(x)$ no es divisible por $1 + x^m$ para $m < 2^r - 1$, entonces $\beta^m \neq 1$ para $m < 2^r - 1$. Como $\beta^j = \beta^i$ para $j > i$, entonces $\beta^j + \beta^i = 0 \leftrightarrow \beta^i(\beta^{j-i} + 1) = 0$. Como estamos en un campo, se tiene $\beta^i = 0$ ó $\beta^{j-i} = 0$; pero $\beta \neq 0$, quiere decir que $\beta^i \neq 0$, por lo que quien deber ser cero es $\beta^{j-i} + 1$, que en forma polinomial sería:

$$\beta^{j-i} = x^{j-i} mód\ h(x)$$
$$\leftrightarrow \beta^{j-i} + 1 = (x^{j-i} + 1) mód\ h(x)$$
$$0 = (x^{j-i} + 1)\ mód\ h(x)$$

Lo cual quiere decir que: $j - i \geq 2^r - 1$, para $j > i \geq 0$, lo que es lo mismo decir que si $j - i < 2^n - 1$, entonces $\beta^j \neq \beta^i$ para $j > i \geq 0$. Por lo tanto tenemos que en el conjunto $\{\beta, \beta^2, \beta^3, \ldots, \beta^{2^r-1}\}$ no tiene potencia alguna que se repita y contiene $2^r - 1$ elementos no nulos, lo que podemos realizar una correspondencia uno a uno con $GF(2^r) \setminus \{0\}$ (que también tiene $2^r - 1$ elementos).

Por la correspondencia entre $\{\beta, \beta^2, \beta^3, \ldots, \beta^{2^r-1}\}$ y $GF(2^r) \setminus \{0\}$ por ultimo notemos que existirá una potencia $k \leq 2^r - 1$ de β tal que sea la identidad. Así, para todo i, tal que $1 \leq i \leq 2^r - 1$ y $i \neq k$, tenemos:

$$\beta^k \beta^i = \beta^i$$
$$\beta^{k+i} = \beta^i$$
$$\rightarrow k + i - i \geq 2^r - 1$$
$$k \geq 2^r - 1, \text{ pero } k \leq 2^r - 1.$$
$$\rightarrow k = 2^r - 1$$

Por lo tanto $\beta^k = \beta^{2^r-1} = 1 = \beta^0$, lo que concluimos:

$$GF(2^r) \setminus \{0\} = \{\beta^i / \ i = 0, 1, \ldots, 2^n - 2\}$$

Esto significa que toda palabra distinta de cero en $\mathcal{K}^n$ puede representarse con la misma potencia de β; esta es una propiedad que facilita la multiplicación en este campo.

Tabla 2.1: Construcción de $GF(2^4)$ usando $h(x) = 1 + x + x^4$

palabra	polinomio en x^i mód $h(x)$	potencia de β
0000	0	— - --
1000	1	$\beta^0 = 1$
0100	x	β^1
0010	x^2	β^2
0001	x^3	β^3
1100	$1 + x \equiv x^4$	β^4
0110	$x + x^2 \equiv x^5$	β^5
0011	$x^2 + x^3 \equiv x^6$	β^6
1101	$1 + x + x^3 \equiv x^7$	β^7
1010	$1 + x^2 \equiv x^8$	β^8
0101	$x + x^3 \equiv x^9$	β^9
1110	$1 + x + x^2 \equiv x^{10}$	β^{10}
0111	$x + x^2 + x^3 \equiv x^{11}$	β^{11}
1111	$1 + x + x^2 + x^3 \equiv x^{12}$	β^{12}
1011	$1 + x^2 + x^3 \equiv x^{13}$	β^{13}
1001	$1 + x^3 \equiv x^{14}$	β^{14}

Un elemento $\alpha \in GF(2^r)$ es primitivo si $\alpha^m \neq 1$ para $1 \leq m < 2^r - 1$.

En consecuencia, α es primitivo si toda palabra distinta de cero en $GF(2^r)$ puede ser representada como potencia de α. De la discusión anterior, vemos que para construir $GF(2^r)$ si se usa un polinomio primitivo, donde β siendo la palabra predefinida, concluimos que β es un elemento primitivo.

Ejemplo 2.8. Usando el polinomio primitivo $h(x) = 1 + x + x^4$, construye $GF(2^4)$. Escribimos cada vector como la potencia de $\beta \leftrightarrow x$ mód $h(x)$ (ver Tabla 2.1). Note que $\beta^{15} = 1$.

Para calcular (0110)(1101) = $\beta^5 \cdot \beta^7 = \beta^{12} = 1111$ ya que $(x + x^2)(1 + x + x^3) \equiv x^5 \cdot x^7 \equiv x^{12}$(mód $h(x)$).

2.2 Polinomio Minimal

Recordemos que α, un elemento en un campo F = *GF* (2^r) es llamado una raíz de un polinomio $p(x) \in F[x]$ si y solamente si $p(\alpha)$ = 0. Esto es, si $p(x) = a_0 + a_1 x + \cdots + a_k x^k$ entonces

$$p(\alpha) = a_0 + a_1\alpha + \cdots + a_k\alpha^k$$

Ejemplo 2.9. Sea $p(x)$ = $1+x^3+x^4$, y el elemento primitivo en *GF* (2^4) es β construido usando $h(x) = 1 + x + x^4$ (ver Tabla 2.1).

$$\begin{aligned} p(\beta) = 1 + \beta^3 + \beta^4 &= 1000 + 0001 + 1100 \\ &= 0101 \\ &= \beta^9. \end{aligned}$$

Así ω no es una raíz de $p(x)$. Sin embargo:

$$\begin{aligned} p(\beta^7) &= 1 + (\beta^7)^3 + (\beta^7)^4 \\ &= 1 + \beta^{21} + \beta^{28} \\ &= 1 + \beta^6 + \beta^{13} \text{ (ya que } \beta^{15} = 1) \\ &= 1000 + 0011 + 1011 = 0000 \\ &= 0. \end{aligned}$$

Puesto que $p(\beta^7) = 0$, β^7 es una raíz de $p(x)$. Note que usada la convención que 1 $\leftrightarrow$ 1000 y 0 $\leftrightarrow$ 0000 así como el hecho que $\beta^{15} = 1$. Así $\beta^{21} = \beta^{15}\beta^6 = 1 \cdot \beta^6 = \beta^6$ y $\beta^{28} = \beta^{15} \cdot \beta^{13} = 1 \cdot \beta^{13} = \beta^{13}$

En general, los elementos distintos de cero α en *GF* (2^r) tienen como orden el entero positivo m más pequeño, tal que $\alpha^m = 1$. Sabemos que cualquier α diferente de cero en *GF* (2^r), el orden de α es de orden $m \leq 2^r - 1$. En particular, α en *GF* (2^r), si su orden es $2^r - 1$, es un elemento primitivo.

Definición 2.1. Definimos el *Polinomio minimal* de α, para cualquier elemento α en *GF* (2^r), como el polinomio de grado mínimo en $\mathcal{K}[x]$ y con raíz α. Sea $\mathfrak{m}_\alpha(x)$ el polinomio minimal de α. Tenga en cuenta que si α es de orden m, (es decir, $\alpha^m = 1$) por lo tanto α es una raíz de $1 + x^m$, por lo tanto cada elemento de *GF* (2^r) es una raíz del mismo polinomio en $\mathcal{K}[x]$.

Si queremos encontrar de un elemento de $GF(2^r)$ su polinomio minimal, apoyémonos en algunos factores relacionados con polinomios minimales.

Teorema 2.1. *Sea $\alpha \neq 0$ un elemento de $GF(2^r)$, con polinomio minimal de α, $\mathfrak{m}_\alpha(x)$. Entonces*

- **i.** *$\mathfrak{m}_\alpha(x)$ es irreducible sobre $\mathcal{K}$,*
- **ii.** *si $f(x)$ es cualquier polinomio sobre $\mathcal{K}$ tal que $f(\alpha) = 0$, entonces $\mathfrak{m}_\alpha(x)$ es un factor de $f(x)$,*
- **iii.** *el polinomio minimal es único, y*
- **iv.** *el polinomio minimal $\mathfrak{m}_\alpha(x)$ es un factor de $1 + x^{2^r-1}$.*

Demostración. (i.) Si $\mathfrak{m}_\alpha(x) = g'(x)h(x)$, entonces $\mathfrak{m}_\alpha(x) = 0$ implica que $g'(x)h(x) = 0$. Por lo tanto, $g'(\alpha) = 0$ o $h(\alpha) = 0$. Dado que $\mathfrak{m}_\alpha(x)$ es el polinomio de menor grado tal que $\mathfrak{m}_\alpha(x) = 0$ entonces $g'(x) = 1$ o $h(x) = 1$. Por lo tanto $\mathfrak{m}_\alpha(x)$ es irreducible sobre $\mathcal{K}$.

(ii.) Por el algoritmo de la división

$$f(x) = \mathfrak{m}_\alpha(x)\, g'(x) + r(x),$$

donde $r(x) = 0$ o grado $r(x) <$ grado $\mathfrak{m}_\alpha(x)$. Ahora $f(\alpha) = 0$, puesto que

$$f(\alpha) = \mathfrak{m}_\alpha(\alpha)\, g'(\alpha) + r(\alpha) = 0 \cdot g'(\alpha) + r(\alpha) = r(\alpha)$$

tenemos que $r(\alpha) = 0$. Por la minimalidad del grado de $\mathfrak{m}_\alpha(x)$, $r(x) = 0$. Por lo tanto $f(x) = \mathfrak{m}_\alpha(x)\, q(x)$, y $\mathfrak{m}_\alpha(x)$ es un factor de $f(x)$.

(iii.) Si $\mathfrak{m}'(x)$ también es un polinomio de grado mínimo tal que $\mathfrak{m}'(\alpha) = 0$, de modo que, por la parte (b), $\mathfrak{m}_\alpha(x)$ es un factor de $\mathfrak{m}'(x)$ y $\mathfrak{m}'(x)$ es un factor de $\mathfrak{m}_\alpha(x)$. Por lo tanto $\mathfrak{m}_\alpha(x) = \mathfrak{m}'(x)$, así es único el polinomio minimal.

(iv.) Sea β un elemento primitivo en $GF(2^r)$ y $\alpha = \beta^i$. Entonces $\alpha^{2^r-1} = (\beta^i)^{2^r-1} = (\beta^{2^r-1})^i = 1^i = 1$. Así α es una raíz de $1 + x^{2^r-1}$ y por (b) $\mathfrak{m}_\alpha(x)$ es un factor de $1 + x^{2^r-1}$. □

Encontrar polinomios minimales de α, $\alpha \in GF(2^r)$, se reduce a encontrar una combinación lineal de los vectores $\{1, \alpha, \alpha^2, \ldots, \alpha^r\}$ que suman 0. Ya que en $\mathcal{K}^r$ es dependiente cualquier conjunto de $r + 1$ vectores, teniendo en cuenta que existe tal combinación.

Una vez que hemos construido *GF*(2^r) usando polinomios minimales, es naturalmente conveniente representar $\mathfrak{m}_\alpha(x)$ por $\mathfrak{m}_i(x)$ donde $\alpha = \beta^i$. Esta notación la introducimos en el ejemplo siguiente.

Ejemplo 2.10. Encontrar el polinomio minimal de $\alpha = \beta^3$, $\alpha \in GF(2^4)$ construido usando $h(x) = 1 + x + x^4$ (ver tabla 2.1). Sea $\mathfrak{m}_\alpha(x) = \mathfrak{m}_3(x) = a_0+a_1x+a_2x^2+a_3x^3+a_4x^4$ entonces debemos encontrar los valores para $a_0, a_1, \ldots, a_4 \in \{0, 1\}$. Note,

$$\begin{aligned}\mathfrak{m}_\alpha(\alpha) = 0 &= a_0 1 + a_1\alpha + a_2\alpha^2 + a_3\alpha^3 + a_4\alpha^4 \\ &= a_0\beta^0 + a_1\beta^3 + a_2\beta^6 + a_3\beta^9 + a_4\beta^{12}\end{aligned}$$

así

$$0000 = a_0(1000) + a_1(0001) + a_2(0011) + a_3(0101) + a_4(1111)$$

resolviendo para a_0, a_1, a_2, a_3, a_4 encontramos que

$$a_0 = a_1 = a_2 = a_3 = a_4 = 1$$

y $\mathfrak{m}_\alpha(x) = 1 + x + x^2 + x^3 + x^4$. Las raíces de $\mathfrak{m}_\alpha(x)$ son $\{\alpha, \alpha^2, \alpha^3, \alpha^4\} = \{\beta^3, \beta^6, \beta^9, \beta^{12}\}$, y así $\mathfrak{m}_3(x) = \mathfrak{m}_6(x) = \mathfrak{m}_9(x) = \mathfrak{m}_{12}(x)$ (donde $\mathfrak{m}_i(x)$ denota el polinomio minimal de β^i).

Si el polinomio minimal para todos los elementos en *GF* (2^r) son buscados entonces tenemos otros hechos útiles. Recordemos que $f(x)^2 = f(x^2)$, así

$$\left(\sum_{i=0}^{n} a_i x^i\right)^2 = \sum_{i=0}^{n} a_i^2 (x^i)^2 = \sum_{i=0}^{n} a_i (x^2)^i.$$

Seguimos del hecho que $(a+ b)^2 = a^2 + b^2$ y el hecho que $a_i^2 = a_i$ puesto que $a_i \in \{0, 1\}$. Así si $f(\alpha) = 0$ entonces $f(\alpha^2) = (f(\alpha))^2 = 0$ y así α^2 es también una raíz de $f(x)$. Similarmente $f(\alpha^4) = (f(\alpha^2))^2 = 0$, etc. y así tenemos que si α es una raíz de $f(x)$ también lo son $\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^i}, \ldots$, etc. Con algunos esfuerzos más probaremos que:

Teorema 2.2. *Sea α un elemento en GF*(2^r) *con polinomio minimal* $\mathfrak{m}_\alpha(x)$*, entonces* $\{\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{r-1}}\}$ *es el conjunto de todas las raíces de* $\mathfrak{m}_\alpha(x)$*. En particular, el grado* $(\mathfrak{m}_\alpha(x))$ es $|\{\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{r-1}}\}|$.

Demostración. Se sabe por definición de $\mathfrak{m}_\alpha(x)$ que $\mathfrak{m}_\alpha(\alpha) = 0$, entonces

$$\mathfrak{m}_\alpha\left(\alpha^{2^i}\right) = (\mathfrak{m}_\alpha(\alpha))^{2^i}$$

$$= (0)^{2^i}$$
$$= 0, \text{ para todo } i \in \mathbb{N}.$$

Además, si $\alpha = \beta^i$, donde $\beta \in GF(2^r)$ es un elemento primitivo, entonces:

$$\begin{aligned} \alpha^{2^r} &= \left(\beta^j\right)^{2^r} \\ &= \left(\beta^{2^r-1}.\beta\right)^j \\ &= (1.\beta)^j \\ &= \beta^j \\ &= \alpha \end{aligned}$$

Esto quiere decir que se repite el ciclo nuevamente, por lo que podemos pensar que el conjunto $\{\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{r-1}}\}$ tiene a todas las raíces de $\mathfrak{m}_\alpha(x)$.

Para confirmar esto, supongamos que hay otra raíz, digamos γ, tal que $\mathfrak{m}_\alpha(\gamma) = 0$ y además satisface que $\gamma \notin \{\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{r-1}}\}$.

Por otro lado, el polinomio mínimo de γ es $\mathfrak{m}_\gamma(x)$ y suponiendo que otro polinomio $\mathfrak{m}_\alpha(x)$ existe que como raíz tiene a γ, por el teorema 2.1. (ii) debe suceder que $\mathfrak{m}_\gamma(x)$ divida a $\mathfrak{m}_\alpha(x)$ y además *grad*$(\mathfrak{m}_\gamma(x)) <$ *grad*$(\mathfrak{m}_\alpha(x))$.

Entonces, por el algoritmo de la división, se tiene que, existe $f(x)$ tal que *grad*$(f(x)) <$ *grad*$(\mathfrak{m}_\alpha(x))$.

$$\mathfrak{m}_\alpha(x) = \mathfrak{m}_\gamma(x) f(x).$$

Si $x = \alpha$, entonces

$$\begin{aligned} \mathfrak{m}_\alpha(\alpha) &= \mathfrak{m}_\gamma(\alpha) f(\alpha) \\ 0 &= \mathfrak{m}_\gamma(\alpha) f(\alpha). \end{aligned}$$

Sabemos que $GF(2^r)$ es un campo, por lo tanto $\mathfrak{m}_\gamma(\alpha) = 0$ ó $f(\alpha) = 0$. Pero para los dos casos, no puede existir otro polinomio de menor grado que el de $\mathfrak{m}_\alpha(x)$ la cual tenga a α como raíz, ya que el polinomio mínimo es único como ya sabemos. Con esto el conjunto $\{\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{r-1}}\}$ tiene a todas las raíces de $\mathfrak{m}_\alpha(x)$.

Por otro lado, por definición de la cardinalidad del conjunto $|\{\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{r-1}}\}|$, es sin repetición, por lo tanto:

Grado $(\mathfrak{m}_\alpha(x))$ es $|\{\alpha, \alpha^2, \alpha^4, \ldots, \alpha^{2^{r-1}}\}|$

□

Ejemplo 2.11. Sea $\mathfrak{m}_5(x)$ el polinomio minimal de $\alpha = \beta^5$, $\beta^5 \in GF\ (2^4)$ (ver Tabla2.1). Ya que $\{\alpha, \alpha^2, \alpha^4, \alpha^8\} = \{\beta^5, \beta^{10}\}$ por teorema 2.2 las raíces de $\mathfrak{m}_5(x)$ son β^5 y β^{10} que significan que el grado $(\mathfrak{m}_5(x)) = 2$ (del Teorema 2.2).

Así $\mathfrak{m}_5(x) = a_0 + a_1x + a_2x^2$, por lo tanto

$$\begin{aligned} 0 &= a_0 + a_1\beta^5 + a_2\beta^{10} \\ &= a_0(1000) + a_1(0110) + a_2(1110). \end{aligned}$$

Así $a_0 = a_1 = a_2 = 1$ y $\mathfrak{m}_5(x) = 1 + x + x^2$.

Similarmente podemos encontrar los polinomios minimales del resto de elementos del campo en *GF* (2^4) construidos usando $1 + x + x^4$. Los resultados son resumidos en la siguiente tabla:

Tabla 2.2: Polinomios minimales en $GF\ (2^4)$

elemento de $GF\ (2^4)$	polinomio minimal
0	x
1	$1 + x$
$\beta, \beta^2, \beta^4, \beta^8$	$1 + x + x^4$
$\beta^3, \beta^6, \beta^9, \beta^{12}$	$1 + x + x^2 + x^3 + x^4$
β^5, β^{10}	$1 + x + x^2$
$\beta^7, \beta^{11}, \beta^{13}, \beta^{14}$	$1 + x^3 + x^4$

2.3 Los códigos de Hamming

De los códigos de Hamming sabemos que tienen una ventaja importante de ser códigos completos de corrección de un solo error y de admitir un simple esquema de decodificación.

Definición 2.2. El código de Hamming corrector de un solo error de longitud $n = 2^r - 1$ para cada $r \geq 2$, tiene una matriz H de verificacion de paridad tal que sus filas son todas las palabras $2^r - 1$ distintas de cero de longitud $n = 2^r - 1$.

En esta sección probaremos que existen códigos cíclicos de Hamming de longitud $n = 2^r - 1$ para cada $r \geq 2$. La ventaja adicional de este código es de ser fácil de codificar, lo cual es común a todos los códigos cíclicos.

Si β es un elemento primitivo de $GF(2^r)$, entonces por la definición de potencias de β son todos diferentes. Entonces podemos crear un código Hamming de longitud $n = 2^r - 1$ que tiene una matriz de control de paridad

$$H = \begin{bmatrix} 1 \\ \beta \\ \beta^2 \\ \vdots \\ \beta^{2^r-2} \end{bmatrix}$$

Note que H es una matriz $(2^r - 1) \times r$.

Ejemplo 2.12. Sea $r = 3$, entonces $n = 2^3 - 1 = 7$. La construcción $GF(2^3)$ con $p(x) = 1 + x + x^3$ y $\beta \leftrightarrow 010$ como primitivo. Recordemos que $\beta^i \leftrightarrow x^i$ mód $p(x)$. En consecuencia, la matriz de verificación de paridad para un código Hamming de longitud 7 es

$$\begin{bmatrix} 1 \\ \beta \\ \beta^2 \\ \beta^3 \\ \beta^4 \\ \beta^5 \\ \beta^6 \end{bmatrix} \leftrightarrow \begin{bmatrix} 100 \\ 010 \\ 001 \\ 110 \\ 011 \\ 111 \\ 101 \end{bmatrix}$$

Vendría ser la misma matriz de verificación de paridad del código cíclico con polinomio generador $p(x)$.

Teorema 2.3. Un polinomio primitivo de grado r es el polinomio generador de un código cíclico Hamming de longitud $2^r - 1$.

Demostración. Sea $\mathcal{C}$ un código cíclico de longitud n con polinomio generador $g'(x)$. Sea $\alpha \in GF(2^r)$ una raíz de $g'(x)$. Entonces para todo $c(x) \in \mathcal{C}$, $c(\alpha) = 0$ y, por lo tanto $m_\alpha(x)$ es un divisor de $c(x)$ por el Teorema 2.1(ii). Siempre podemos escribir $g'(x)$ como un producto polinomial mínimo de elementos en $GF(2^r)$. □

Podemos usar este algoritmo de verificación y decodificador de paridad de matriz en C.

Teorema 2.4. *Sea $g'(x)$ el generador para un código cíclico $\mathcal{C}$ de longitud n entonces $g'(x)$ será el producto (mínimo común múltiplo) de polinomios minimales de $\alpha_1, \alpha_2, \ldots, \alpha_k \in GF(2^r)$, con α_i una raíz de $1 + x^n$, si y solamente si para todo $c(x) \in \mathcal{C}$*

$$c(\alpha_1) = c(\alpha_2) = \cdots = c(\alpha) = 0.$$

Demostración. La decodificación de códigos cíclicos de Hamming es fácil. Si el generador es un polinomio primitivo $m_\alpha(x)$, y $w(x)$ es recibido, entonces $w(x) = c(x) + e(x)$, $c(x) \in \mathcal{C}$ y $w(\alpha) = e(\alpha) = \alpha^j$. Por lo tanto, el polinomio error más probable es $e(x) = x^j$, por lo que $c(x) = w(x) + x^j$. □

Ejemplo 2.13. Supónga que $GF(2^3)$ fue construido usando $1 + x + x^3$. Entonces $m_1(x) = 1 + x + x^3$ es un generador para un código cíclico Hamming de longitud 7. Supónga que $w(x) = 1 + x + x^3 + x^6$ es recibido. Entonces

$$\begin{aligned} w(\beta) &= 1 + \beta^2 + \beta^3 + \beta^6 \\ &= 100 + 001 + 110 + 101 \\ &= 110 \\ &= \beta^3 \end{aligned}$$

Así $e(x) = x^3$ y $c(x) = w(x) + x^3 = 1 + x^2 + x^6$.

2.4 Códigos BCH

Los *códigos Bose-Chaudhuri-Hocquengham*, o códigos BCH son una clase importante de códigos de corrección de errores multiples. El procedimiento para construir y decodificar códigos BCH se desarrollará más adelante. Construiremos primero y un ejemplo importante de clases decodificaremos, para saber la familia de códigos correctores de dos error BCH.

El código BCH es importante por dos razones. En primer lugar, admiten esquemas de decodificación relativamente simples y, en segundo lugar, la categoría de códigos BCH es muy amplia. Por supuesto, para todos los enteros positivos r y t con $t \leq 2^{r-1} - 1$, existe un código BCH de longitud $n = 2^r - 1$ que es el corrector de t error y tiene dimensión $k \geq n - rt$.

Definición 2.3. Los Códigos correctores de 2 error BCH de longitud $2^r - 1$ son códigos lineales cíclicos generados por $g'(x) = m_\beta(x)m_{\beta^3}(x)$, donde β es el elemento primitivo en $GF(2^r)$ y $r \geq 4$.

Dado que $n = 2^r - 1$ y $g'(x)$ divide $1 + x^n$ (por Teorema 2.1(iii)) $g'(x)$ es un polinomio generador para el código cíclico.

Ejemplo 2.14. β es un elemento primitivo en $GF(2^4)$ construido con $p(x) = 1 + x + x^4$ (ver Tabla 2.1). Tenemos que $m_1(x) = 1 + x + x^4$ y $m_3(x) = 1 + x + x^2 + x^3 + x^4$. Por lo que tenemos

$$g'(x) = m_1(x)m_3(x) = 1 + x^4 + x^6 + x^7 + x^8$$

es un generador para el código corrector de 2 error BCH de longitud 15.

Lema 2.1. La siguiente matriz H es una matriz de verificación de paridad para el código corrector de 2 error BCH de longitud $2^r - 1$, donde β es un elemento primitivo en $GF(2^r)$ y el polinomio generador es $g'(x) = m_1(x)m_3(x)$

$$H = \begin{bmatrix} \beta^0 & \beta^0 \\ \beta & \beta^3 \\ \beta^2 & \beta^6 \\ \vdots & \vdots \\ \beta^i & \beta^{3i} \\ \vdots & \vdots \\ \beta^{2^r-2} & \beta^{3(2^r-2)} \end{bmatrix}$$

Demostración. Puesto que β^i es un elemento de *GF* (2^r), representa una palabra de longitud r, así H es una matriz $(2^r - 1) \times (2r)$. Como el $grado(\mathfrak{m}_1(x)) = r = grado(\mathfrak{m}_3(x))$, el grado de $g'(x) = \mathfrak{m}_1(x)\mathfrak{m}_3(x)$ es $2r$, por lo que la dimensión del código $n - 2r = 2^r - 1 - 2r$. □

Pongamos por caso, en la Tabla 2.1 usamos $GF(2^4)$ construido con el polinomio primitivo $p(x) = 1 + x + x^4$ para construir un código corrector de 2 error BCH $\mathcal{C}_{15}$. El código $\mathcal{C}_{15}$ se define como el código lineal con matriz verificadora de paridad H 15×8, y $\mathfrak{m}_1(x)\mathfrak{m}_3(x)$ como polinomio generador.

Tabla 2.3: La matriz verificadora de paridad de $\mathcal{C}_{15}$

1	1	1000	1000
β	β^3	0100	0001
β^2	β^6	0010	0011
β^3	β^9	0001	0101
β^4	β^{12}	1100	1111
β^5	1	0110	1000
β^6	β^3	0011	0001
β^7	β^6	1101	0011
β^8	β^9	1010	0101
β^9	β^{12}	0101	1111
β^{10}	1	1110	1000
β^{11}	β^3	0111	0001
β^{12}	β^6	1111	0011
β^{13}	β^9	1011	0101
β^{14}	β^{12}	1001	1111

Teorema 2.5. *Para cualquier entero* $r \geq 4$; *existe un código corrector de* 2 *error* BCH *de longitud* $n = 2^r - 1$, *dimensión* $k = 2^r - 2r - 1$ *y distancia* $d = 5$ *con polinomio generador* $\mathfrak{m}_1(x)\mathfrak{m}_3(x)$.

Demostración. Para probar que la distancia es 5, mostraremos que este puede corregir 2 error y así tiene distancia sea al menos 5. Está claro a partir de la definición de la matriz de control de paridad, que $n = 2^r - 1$, y puesto que $\mathfrak{m}_1(x)$ y $\mathfrak{m}_3(x)$ tienen grado r, $g'(x)$ tiene grado $n - k = 2r$ entonces $k = 2^r - 2r - 1$ □

2.5 Decodificando códigos corrector de 2 error BCH

Describimos el escenario de decodificación para códigos corrector de 2 error BCH creado en la sesión anterior. En esta sección, definiremos una palabra binaria cuya longitud *r* es igual a la potencia de β.

Una matriz de verificación de paridad para el $(2^r - 1, 2^r - 2r - 1)$ código corrector de 2 error BCH con generador $g'(x) = m_1(x)m_3(x)$ en H definida como en el Lema 2.1.

Supongamos que se recibe una palabra w y $w \leftrightarrow w(x)$. Así que el síndrome de w es

$$wH = [w(\beta), w(\beta^3)] = [s_1, s_3]$$

donde s_1 y s_3 son palabras de longitud r.

Si en la transmisión no ocurre ningún error, entonces el síndrome es $wH = 0$, así $s_1 = s_3 = 0$. Si en la transmisión ocurre sólo un único error, entonces, el error polinómico es $e(x) = x^i$ así $wH = eH = [e(\beta), e(\beta^3)] = [s_1, s_3]$. Por lo tanto $s_1^3 = s_3$.

Si en la transmisión ocurren dos errores, en las posiciones i y j, con $i \neq j$, entonces $e(x) = x^i + x^j$ y $wH = eH = [e(\beta), e(\beta^3)] = [s_1, s_3]$. Así el síndrome wH es dado por

$$wH = [s_1, s_3] = [\beta^i + \beta^j, \beta^{3i} + \beta^{3j}].$$

Consideremos el siguiente sistema de ecuaciones

$$\begin{aligned} \beta^i + \beta^j &= s_1 \\ \beta^{3i} + \beta^{3j} &= s_3. \end{aligned}$$

Ahora tenemos la factorización

$$(\beta^i + \beta^j)(\beta^{2i} + \beta^{i+j} + \beta^{2j}) = \beta^{3i} + \beta^{3j},$$

y

$$s_1^2 = (\beta^i + \beta^j)^2 = \beta^{2i} + \beta^{2j}.$$

Por lo tanto

$$\begin{aligned} s_3 &= \beta^{3i} + \beta^{3j} \\ &= (\beta^i + \beta^j)(\beta^{2i} + \beta^{2j} + \beta^{i+j}) \\ &= s_1(s_1^2 + \beta^{i+j}). \end{aligned}$$

Así

$$\frac{s_3}{s_1} + s_1^2 = \beta^{i+j}$$

Ahora β^i y β^j son raíces de la ecuación cuadrática

$$x^2 + (\beta^i + \beta^j)x + \beta^{i+j} = 0$$

y de aquí las raíces de

$$x^2 + s_1 x + \left(\frac{s_3}{s_1} + s_1^2\right) = 0$$

Por lo tanto, al resolver esta ecuación, podemos encontrar la ubicación de la falla. El polinomio del lado izquierdo de esta ecuación se llama polinomio localizador de error.

Ejemplo 2.15. Sea $w \leftrightarrow w(x)$ la palabra recibida con síndromes s_1 = 0111 = $w(\beta)$ y s_3 = 1010 = $w(\beta^3)$, donde w fue encodificada usando $\mathcal{C}_{15}$. De la Tabla 2.1 tenemos que $s_1 \leftrightarrow \beta^{11}$ y $s_3 \leftrightarrow \beta^8$. Entonces

$$\begin{aligned}\frac{s_3}{s_1} + s_1^2 &= \beta^8\beta^{-11} + \beta^{22} \\ &= \beta^{12} + \beta^7 \\ &= \beta^2.\end{aligned}$$

Formamos el polinomio $x^2 + \beta^{11}x + \beta^2$ y descubrimos que tiene raíces β^4 y β^{13}. En consecuencia, podemos decidir que en las posiciones 4 y 13 ocurren los errores más probables (es decir, $e(x) = x^4 + x^{13}$) de modo que el patrón de error más probable es

000010000000010.

Hemos llegado a una escena para la decodificación incompleta de maxima probabilidad para el código corrector de 2 error BCH. Sea w una palabra recibida. Una vez que se identifica el patrón de error, el algoritmo, por supuesto, está completo.

Algoritmo 2.1. Decodificación de máxima probabilidad incompleta para el código corrector de 2 error BCH con el polinomio generador $\mathfrak{m}_1(x)\mathfrak{m}_3(x)$.

1. Calcular el síndrome wH = [s_1, s_3] = [$w(\beta)$, $w(\beta^3)$].
2. Si $s_1 = s_3 = 0$, concluimos que ningún error a ocurrido. Decodifique $c = w$ como la palabra código enviada.

3. Si $s_1 = 0$ y $s_3 \neq 0$ solicitamos una retransmisión.
4. Si $s_1^3 = s_3$ entonces corrige un simple error en la posición $\boldsymbol{i}$, donde $s_1 = \beta^i$.
5. De la ecuación cuadrática

$$x^2 + s_1 x + \frac{s_3}{s_1} + s_1^2 = 0 \qquad (*)$$

6. Si la ecuación (*) tiene dos raíces distintas β^i y β^j, corrige el error de posición i y j.
7. Si la ecuación (*) no tiene dos raíces distintas en $GF(2^r)$, concluimos que existió tres errores como mínimo en la transmisión, y solicitamos una retransmisión.

A continuación los ejemplos que siguen, usan $\mathcal{C}_{15}$ cuya matriz verificadora de paridad estalistada en la Tabla 2.3 y polinomio generador $g'(x)$ listado en el ejemplo 2.15.

Ejemplo 2.16. Suponemos que se recibe w y el síndrome es wH = 01111010 ↔ [β^{11}, β^8]. Ahora

$$s_1^3 = (\beta^{11})^3 = \beta^{33} = \beta^3 \neq \beta^8 = s_3.$$

En este caso la ecuación (*) es $x^2 + \beta^{11}x + \beta^2 = 0$, como es mostrado en el ejemplo 2.16. Las raíces de esta ecuacion β^4 y β^{13}. Por lo tanto, corregimos los errores en las posiciones $i = 4$ y $j = 13$; en otras palabras, el más posible patrón de error es u = 000010000000010, y $e(x) = x^4 + x^{13}$ es el polinomio error presumido.

Ejemplo 2.17. Asuma que el síndrome es wH = [$w(\beta)$, $w(\beta^3)$] = [β^3, β^9]. Entonces $s_1^3 = (\beta^3)^3 = \beta^9 = s_3$. Por lo tanto, es más probable que haya ocurrido un error singular en la posición $i = 3$. El error más posible es u = 000100000000000, y $e(x) = x^3$ es el polinomio error.

Ejemplo 2.18. Asuma que w = 110111101011000 es recibido. El síndrome es

$$wH = 01110110 \leftrightarrow [\beta^{11}, \beta^5] = [s_1, s_3].$$

Ahora $s_1^3 = \beta^{33} = \beta^3 \neq s_3 = \beta^5$. Para formar la ecuación cuadrática (*), primero calculamos

$$\begin{aligned} \frac{s_3}{s_1} + s_1^2 &= \beta^5\beta^{-11} + (\beta^{11})^2 \\ &= \beta^9 + \beta^7 \\ &\leftrightarrow 0101 + 1101 \\ &= 1000 \\ &\leftrightarrow \beta^0. \end{aligned}$$

Así en este caso, (*) será

$$x^2 + \beta^{11}x + \beta^0 = 0.$$

Probando los elementos de $GF(2^4)$ como las posibles raíces, llegamos a $x = \beta^7$ y encontramos

$$\begin{aligned}(\beta^7)^2 + \beta^{11}\beta^7 + \beta^0 &= \beta^{14} + \beta^3 + \beta^0 \\ &\leftrightarrow 1001 + 0001 + 1000 \\ &= 0000.\end{aligned}$$

Ahora $\beta^7\beta^j = 1 = \beta^{15}$, así $\beta^j = \beta^8$ es otra raíz. Entonces, en las posiciones $i = 7$ y $j = 8$, corregimos errores; esto es $u = 000000011000000$ es más probable patrón de error, decodificamos $v = w + u = 110111110011000$ como la palabra que se envió.

Ejemplo 2.19. Digamos que de $\mathcal{C}_{15}$ una palabra es enviada, por lo que ocurren errores en las posiciones 2, 6 y 12. Por lo tanto, el síndrome wH es la suma de las filas 2, 6 y 12 de H, donde w es la palabra enviada. Entonces

$$\begin{aligned}wH &= 00100011 + 00110001 + 11110011 \\ &= 11100001 \leftrightarrow [\beta^{10}, \beta^3] = [s_1, s_3].\end{aligned}$$

Ahora $s_1^3 = (\beta^{10})^3 = \beta^{30} = 1 \neq \beta^3$. Calculamos

$$\begin{aligned}\frac{s_3}{s_1} + s_1^2 &= \beta^3\beta^{-10} + \beta^{20} \\ &\leftrightarrow 1010 + 0110 = 1100 \leftrightarrow \beta^4\end{aligned}$$

y entonces la ecuación cuadrática es

$$x^2 + \beta^{10}x + \beta^4 = 0.$$

Probando cada elemento de $GF(2^4)$, observamos que esta ecuación no tiene raíces en $GF(2^4)$. En consecuencia, la decodificación de máxima probabilidad es incompleta para que $\mathcal{C}_{15}$ concluya correctamente que se han producido al menos tres errores y debemos retransmitir.

Capítulo 3

Aplicación de los Códigos BCH

Tal y como vimos en teoría, un problema fundamental a la hora de enviar mensajes es cómo transmitir información a través de un canal con ruido de forma que el receptor sepa detectar si el mensaje que le hemos enviado es correcto o contiene errores.
La idea clave consiste en codificar la información (binaria), añadiéndole redundancia que nos permita recuperar la información que aún en el caso de que haya algún error en la transmisión.

3.1 Codificación de mensajes de texto

El proceso de codificación de un mensaje de texto consiste en asignar una letra a cada palabra de $\mathcal{K}^{15}$ luego multiplicar éste código por una matriz generadora, dicho producto del código por la matriz generadora es el mensaje a transmitir.

Ejemplo 3.1. Supongamos que queremos transmitir el mensaje: $HELP$. Para tal fin asignamos una palabra a cada letra:

$$H = 110000000000000$$
$$E = 000010000100001$$
$$L = 010001010100000$$
$$P = 110011100111000$$

Ahora codificamos el mensaje usando la matriz de verificación de paridad que se encuentra en la Tabla 2.3, así tenemos

- $\mathcal{C}(H) = [110000000000000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [1100 \;\; 1001]$

- $\mathcal{C}(E) = [000010000100001] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0000 \;\; 1111]$

- $\mathcal{C}(L) = [0100010101000000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [1010\ \ 0101]$

- $\mathcal{C}(P) = [1100000000000000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0000\ \ 0111]$

Luego el mensaje codificado es

$$\begin{aligned} \mathcal{C}(H) &= [11001001] \\ \mathcal{C}(E) &= [00001111] \\ \mathcal{C}(L) &= [10100101] \\ \mathcal{C}(P) &= [00000111] \end{aligned}$$

Ejemplo 3.2. Supongamos que queremos transmitir el mensaje: *HOLA*. Para tal fin asignamos una palabra a cada letra:

$$\begin{aligned} H &= 100000000000000 \\ O &= 000000000000001 \\ L &= 100001000010000 \\ A &= 000010000100001 \end{aligned}$$

Ahora codificamos el mensaje usando la matriz de verificación de paridad que se encuentra en la Tabla 2.3, así tenemos

- $\mathcal{C}(H) = [100000000000000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [1000 \ \ 1000]$

- $\mathcal{C}(O) = [000000000000001] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [1001 \;\; 1111]$

- $\mathcal{C}(L) = [100001000010000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0000 \;\; 1100]$

- $\mathcal{C}(A) = [000010000100001] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0000 \ \ 1111]$

Luego el mensaje codificado es

$$\mathcal{C}(H) = [10001000]$$
$$\mathcal{C}(O) = [10011111]$$
$$\mathcal{C}(L) = [00001100]$$
$$\mathcal{C}(A) = [00001111]$$

Ejemplo 3.3. Supongamos que queremos transmitir el mensaje: "*PONER FÁCIL",* sin considerar espacios ni acentos. Para tal fin asignamos una palabra a cada letra:

$$P = 110011100100000$$
$$O = 111000000000001$$
$$N = 101110000000000$$
$$E = 010000100000001$$
$$R = 010101001011000$$

$$F = 110111010001100$$
$$A = 101110000001000$$
$$C = 000010000100001$$
$$I = 111001011000000$$
$$L = 000111010000110$$

Ahora codificamos el mensaje usando la matriz de verificación de paridad que se encuentra en la Tabla 2.3, así tenemos

- $\mathcal{C}(P) = [110011100100000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0000 \;\; 0000]$

- $\mathcal{C}(O) = [111000000000001] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0111 \;\; 0101]$

- $\mathcal{C}(N) = [101110000000000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0111 \;\; 0001]$

- $\mathcal{C}(E) = [010000100000001] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [1110 \;\; 1111]$

- $\mathcal{C}(R) = [010101001011000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0000 \;\; 0000]$

- $\mathcal{C}(F) = [110111010001100] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0010\ \ 1010]$

- $\mathcal{C}(A) = [101110000001000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0000\ \ 0000]$

- $\mathcal{C}(C) = [000010000100001] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0000\ \ 1111]$

- $\mathcal{C}(I) = [111001011000000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [1111\ \ 0100]$

- $\mathcal{C}(L) = [000111010000110] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0010\ \ 0111]$

Luego el mensaje codificado es

$$\mathcal{C}(P) = [00000000]$$
$$\mathcal{C}(O) = [01110101]$$
$$\mathcal{C}(N) = [01110001]$$
$$\mathcal{C}(E) = [11101111]$$
$$\mathcal{C}(R) = [00000000]$$
$$\mathcal{C}(F) = [00100111]$$
$$\mathcal{C}(A) = [00000000]$$
$$\mathcal{C}(C) = [00001111]$$
$$\mathcal{C}(I) = [11110100]$$
$$\mathcal{C}(L) = [00100111]$$

Ejemplo 3.4. Supongamos que queremos transmitir el mensaje: "*HÁGALO RÁPIDO",* sin considerar espacios ni acentos. Para tal fin asignamos una palabra a cada letra, en esta oportunidad consideramos una sola palabra para cada letra:

$$H = 100100000100000$$
$$A = 000010000000001$$
$$G = 101010000000000$$
$$A = 000010000000001$$
$$L = 010100100000001$$
$$O = 011100001111000$$

$$R = 110100010001001$$
$$A = 000010000000001$$
$$P = 111110000100001$$
$$I = 011000000000000$$
$$D = 000001000000110$$
$$O = 011100001111000$$

Ahora codificamos el mensaje usando la matriz de verificación de paridad que se encuentra en la Tabla 2.3, así tenemos

- $\mathcal{C}(H) = [100100000100000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0100 \ \ 0010]$

- $\mathcal{C}(A) = [000010000000001] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [1000\ \ 1010]$

- $\mathcal{C}(G) = [101010000000000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0110\ \ 0100]$

- $\mathcal{C}(L) = [010100100000001] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [1111\ \ 1010]$

- $\mathcal{C}(O) = [011100001111000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [1001\ \ 0100]$

- $\mathcal{C}(R) = [110100010001001] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [1110\ \ 0001]$

- $\mathcal{C}(P) = [111110000100001] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [1111\ \ 0000]$

- $\mathcal{C}(I) = [011000000000000] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0110\ \ 0010]$

- $\mathcal{C}(D) = [000001000000110] \begin{bmatrix} 1000 & 1000 \\ 0100 & 0001 \\ 0010 & 0011 \\ 0001 & 0101 \\ 1100 & 1111 \\ 0110 & 1000 \\ 0011 & 0001 \\ 1101 & 0011 \\ 1010 & 0101 \\ 0101 & 1111 \\ 1110 & 1000 \\ 0111 & 0001 \\ 1111 & 0011 \\ 1011 & 0101 \\ 1001 & 1111 \end{bmatrix} = [0010\ \ 1110]$

Luego el mensaje codificado es

$$\begin{aligned} \mathcal{C}(H) &= [01000010] \\ \mathcal{C}(A) &= [10001010] \\ \mathcal{C}(G) &= [01100100] \\ \mathcal{C}(A) &= [10001010] \\ \mathcal{C}(L) &= [11111010] \\ \mathcal{C}(O) &= [10010100] \\ \mathcal{C}(R) &= [11100001] \\ \mathcal{C}(A) &= [10001010] \end{aligned}$$

$$\mathcal{C}(P) = [11110000]$$
$$\mathcal{C}(I) = [01100010]$$
$$\mathcal{C}(D) = [00101110]$$
$$\mathcal{C}(O) = [10010100]$$

3.2 Localización y corrección de error

Ahora veamos si ha ocurrido un error en la transmisión de cada una de las palabras código y localicemoslo.

1. Para el primer ejemplo se tiene:

$$\mathcal{C}(H) = [11001001] \leftrightarrow [\beta^4, \beta^{14}] = [s_1, s_3]$$

Ahora $s^3 = \beta^{12} \neq s_3 = \beta^{14}$. Calculamos

$$\frac{s_3}{s_1} + s_1^2 = \beta^{14}\beta^{-4} + (\beta^4)^2$$
$$= \beta^{10} + \beta^8$$
$$\leftrightarrow 1110 + 1010$$
$$= 0100$$
$$\leftrightarrow \beta^1$$

Así en este caso, la ecuación cuadrática será

$$x^2 + \beta^4 x + \beta^1 = 0$$

Probando los elementos de $GF(2^4)$ como las posibles raíces, llegamos a $x = \beta^1$

$$(\beta^1)^2 + \beta^4\beta^1 + \beta^1 = \beta^2 + \beta^5 + \beta^1$$
$$\leftrightarrow 0010 + 0110 + 0100$$
$$= 0000$$

Ahora $\beta^1\beta^j = \beta$ lo que implica que $\beta^j = 1 = \beta^{15}$, así $\beta^j = \beta^{15}$ es la otra raíz. Por lo tanto corregimos errores en las posiciones 1 y 15; es decir $u = 010000000000001$. Luego decodificamos $v = w + u = 100000000000001$ como la palabra enviada.

- Para la letra que sigue, tenemos

$$\mathcal{C}(E) = [00001111] \leftrightarrow [\text{--}, \beta^{12}] = [s_1, s_3]$$

$$S_1 = 0 \quad \text{y} \quad S_3 = \beta^{12} \neq 0$$

Por lo tanto, solicitamos una retransmisión.

- Para la siguiente letra, tenemos

$$\mathcal{C}(L) = [10100101] \leftrightarrow [\beta^8, \beta^9] = [s_1, s_3]$$

Ahora $s_1^3 = (\beta^8)^3 = \beta^{24} = \beta^{15}.\beta^9 = \beta^9 = s_3$

Por lo tanto es más probable que haya ocurrido un error singular en la posición $i = 8$, es decir u = 000000001000000. Luego decodificamos $v = w + u$ = 010001011100000 como la palabra enviada.

- Y por último, tenemos

$$\mathcal{C}(P) = [00000111] \leftrightarrow [\text{--}, \beta^{11}] = [s_1, s_3]$$

$$S_1 = 0 \quad \text{y} \quad S_3 = \beta^{11} \neq 0$$

Por lo tanto, solicitamos una retransmisión.

2. Para el segundo ejemplo se tiene:

$$\mathcal{C}(H) = [10001000] \leftrightarrow [1,1] = [s_1, s_3]$$

Ahora como $s_1 = 1000$ y $s_3 = 1000$, entonces $s_1 = s_3$ lo que indica que ningún error ha ocurrido durante la transmisión.

- Para la siguiente letra, se tiene

$$\mathcal{C}(O) = [10011111] \leftrightarrow [\beta^{14}, \beta^{12}] = [s_1, s_3]$$

Ahora $s_1^3 = (\beta^{14})^3 = \beta^{42} = \beta^{15}.\beta^{15}.\beta^{12} = \beta^{12} = s_3$

Por lo tanto es más probable que haya ocurrido un error singular en la posición $i = 12$, es decir u = 000000000000100. Luego decodificamos $v = w + u$ = 000000000000101 como la palabra enviada.

- Para la letra que sigue, tenemos

$$\mathcal{C}(L) = [00001100] \leftrightarrow [--, \beta^4] = [s_1, s_3]$$

$$S_1 = 0 \quad \text{y} \quad S_3 = \beta^4 \neq 0$$

Por lo tanto, solicitamos una retransmisión.

- Para la última letra, tenemos

$$\mathcal{C}(A) = [00001111] \leftrightarrow [--, \beta^{12}] = [s_1, s_3]$$

$$S_1 = 0 \quad \text{y} \quad S_3 = \beta^{12} \neq 0$$

Por lo tanto, solicitamos una retransmisión.

3. Para el tercer ejemplo se tiene:

$$\mathcal{C}(H) = [00000000] \leftrightarrow [s_1, s_3]$$

Ahora como $s_1 = 0000$ y $s_3 = 0000$, entonces $s_1 = s_3$ lo que indica que ningún error ha ocurrido durante la transmisión.

Para las demás letras del tercer y cuarto ejemplo se procede de la misma manera.

Conclusiones

1. En esta tesis se localizó y corrigió errores en la transmisión de mensajes haciendo uso de los códigos BCH, porque es una clase de códigos de corrección de errores dobles la cual nos permite la mejora de envío y recepción de información.

2. Se encontró la matriz de verificación de paridad para los códigos correctores de 2 error BCH, que tiene como filas palabras que se encuentran en la tabla 2.1.

3. Se encontró el código $\mathcal{C}_{15}$, que tiene como matriz verificadora de paridad de la tabla 2.3 y polinomio generador $\mathfrak{m}_1(x)\mathfrak{m}_3(x)$.

4. Para el proceso de codificación se formó el código $\mathcal{C}_{15}$ que resulta del producto de la palabra por una de las columnas de la tabla 2.3.

5. Tanto para el proceso de codificación y de la localización y corrección de error se utilizó la tabla 2.3.

Sugerencias

- Con el avanzar de la ciencia y la tecnología, la matemática va evolucionando en diferentes ciencias, debido a ello sería de gran ayuda un curso sobre la Teoría de Códigos.

- Para futuros investigadores, pueden considerar este estudio y quizá aclarar o profundizar en muchos puntos que pueden haberse quedado vacíos.

- Para que el trabajo realizado en la práctica sea aplicado en el campo de las telecomunicaciones e informática, con el objetivo de mejorar la trasmisión de mensajes detectando y corrigiendo errores, producidos a causa del ruido.

Bibliografía

[1] **Devaud Gloria; Erpendilng María; Kirsten Lilian; Navarro María; Ortega Myriam; Vicente Miryam**. "*Álgebra Lineal*". Universidad de Concepción.

[2] **Godement**; "*Álgebra*"; Editorial: Tecnos, España - 1978.

[3] **Grimaldi**;"*Matemática Discreta y Combinatoria*"; Editorial Adisson-Wesley, Mexico-1998.

[4] **Hoffman D.G.**."*Coding Theory: The Essentials*". Marcel Dekker.Inc, 1992.

[5] **Inchausti**;"*Matemática(Análisis)*"; Editorial NR,España-1979.

[6] **Johnson Baugh**;"*Matemáticas Discretas*"; Editorial Prentice-Hall, Mexico-1999.

[7] **K. P. Bogart**; "*Introductory Combinatorics, 2nd ed.*"; Harcourt Brace Jovanovich, San Diego, CA, 1990.

[8] **Lang Serge**; "*Álgebra*; Editorial: Aguilar España 1971.

[9] **Pita Ruiz**. "*Álgebra Lineal*". Editorial Prentice Hall Hispanoamericana S.A.

[10] **Ralph P. Grimaldi** . "*Discrete and Combinatorial Mathematics an Applied Introduction Third Edition*". Addison-Wesley Publishing Company.

[11] **R. A. Brualdi**; "*Introductory Combinatorics, 3rd ed.*"; Prentice-Hall, Upper Saddle River, NJ, 1999.

[12] **Russel Merris**; "*Combinatorics, segunda edición*; John Wiley & Sons, Inc, 2003.

[13] **Todd K. Moon**. "*Error Correction Coding Mathematical Methods and Algorithms*". John Wiley & SONS, INC.

[14] **Valero Elizondo L.**; "*Notas del Curso de Álgebra Moderna III*"; Michoacán. México, Marzo del 2005.

UNIVERSIDAD NACIONAL PEDRO RUIZ GALLO
FACULTAD DE CIENCIAS FISICAS Y MATEMATICAS
D E C A N A T O
Ciudad Universitaria - Lambayeque

ACTA DE SUSTENTACIÓN VIRTUAL Nº 014-2023-D/FACFyM

Siendo las 9:00 am del día 20 de marzo del 2023, se reunieron vía plataforma virtual, https://meet.google.com/yep-ucdv-ggj?authuser=0 los miembros del jurado evaluador de la Tesis titulada:

"**LOCALIZACIÓN Y CORRECCIÓN DE ERRORES EN LA TRANSMISIÓN DE MENSAJES MEDIANTE CÓDIGOS BCH**".

Designados por Resolución N° 932-2021-VIRTUAL-D/FACFyM de fecha 20 de diciembre de 2021. Con la finalidad de evaluar y calificar la sustentación de la tesis antes mencionada, conformada por los siguientes docentes:

Dr. Lic. Mat. Segundo Leonardo Valdivia Velásquez	Presidente
Dr. Lic. Mat. Rubén Esteban Burga Barboza	Secretario
Lic. Mat. Miguel Ángel Baca Ferreyros	Vocal

La tesis fue asesorada por el Dr. Walter Arriaga Delgado, nombrado por Resolución N° 932-2021-VIRTUAL- D/FACFyM de fecha 20 de diciembre de 2021.

El Acto de Sustentación fue autorizado por Resolución N° 212-2023 -VIRTUAL-D/FACFyM de fecha 9 de marzo de 2023. La Tesis fue presentada y sustentada por el Bachiller: Oliva Suarez Eduardo Enrique y tuvo una duración de 30 minutos.

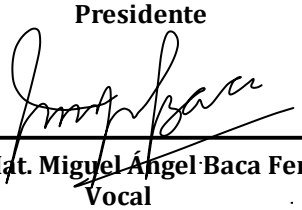
Después de la sustentación, y absueltas las preguntas y observaciones de los miembros del jurado se procedió a la calificación respectiva, otorgándole el Calificativo de 16 (dieciséis) en la escala vigesimal, mención Bueno.

Por lo que queda apto para obtener el Título Profesional de **Licenciado en Matemáticas**, de acuerdo con la Ley Universitaria 30220 y la normatividad vigente de la Facultad de Ciencias Físicas y Matemáticas y la Universidad Nacional Pedro Ruiz Gallo.

Siendo las 10:01 am se dio por concluido el presente acto académico, dándose conformidad al presente acto con la firma de los miembros del jurado.

Dr. Lic. Mat. Segundo leonardo Valdivia Velásquez
Presidente

Dr. Lic. Mat. Rubén Esteban Burga Barboza
Secretario



Lic. Mat. Miguel Ángel Baca Ferreyros
Vocal

Dr. Lic. Mat. Walter Arriaga Delgado
Asesor

CONSTANCIA DE VERIFICACIÓN DE ORIGINALIDAD

(RESOLUCIÓN N° 626-2021-CU DEL 30 DE DICIEMBRE 2021)

Yo, Walter Arriaga Delgado, usuario revisor del documento titulado: Localización y corrección de errores en la transmisión de mensajes mediante códigos BCH, cuyo autor es, Eduardo Enrique Oliva Suárez, de la Escuela Profesional de Matemática de la Facultad de Ciencias Físicas y Matemáticas de la Universidad Nacional Pedro Ruiz Gallo, luego de la revisión exhaustiva del documento de tesis constato que la misma tiene un índice de similitud del 18% verificable en el reporte de similitud del programa Turnitin.

El suscrito analizó dicho reporte y concluyó que cada una de las coincidencias detectadas dentro del porcentaje de similitud permitido no constituyen plagio y que el documento cumple con la integridad científica y con las normas para el uso de citas y referencias establecidas en los protocolos respectivos.

Se cumple con adjuntar el Recibo Digital a efectos de la trazabilidad respectiva del proceso.

Lambayeque, 03 de marzo de 2023

Dr. Walter Arriaga Delgado
DNI: 16732082
ASESOR

REPORTE AUTOMATIZADO DE SIMILITUDES

Final-final

INFORME DE ORIGINALIDAD

18%	18%	2%	3%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	investigaciones.uniatlantico.edu.co Fuente de Internet	8%
2	ri.ues.edu.sv Fuente de Internet	5%
3	Sacristán Riquelme, Jordi. "Sistema implantable para la estimulación y registro de nervio periférico", Bellaterra : Universitat Autònoma de Barcelona,, 2007 Fuente de Internet	1%
4	Submitted to Universitat Politècnica de València Trabajo del estudiante	1%
5	Svozil, K.. "Computational universes", Chaos, Solitons and Fractals, 200508 Publicación	1%
6	ddfe.curtin.edu.au Fuente de Internet	<1%
7	www.fi-b.unam.mx Fuente de Internet	<1%

Dr. Lic. Mat. Walter Arriaga Delgado
DNI: 16732082
Asesor

8	Submitted to Universidad Nacional Pedro Ruiz Gallo Trabajo del estudiante	<1 %
9	core.ac.uk Fuente de Internet	<1 %
10	jason.thanez.net Fuente de Internet	<1 %
11	repositorio.unprg.edu.pe Fuente de Internet	<1 %
12	Submitted to Auckland International College Trabajo del estudiante	<1 %
13	repositorio.unprg.edu.pe:8080 Fuente de Internet	<1 %
14	Submitted to Higher Education Commission Pakistan Trabajo del estudiante	<1 %
15	Levesque, . "Single Core Optimization", Chapman & Hall/CRC Computational Science, 2010. Publicación	<1 %
16	allasm.com Fuente de Internet	<1 %
17	Submitted to Kings Christian College Trabajo del estudiante	<1 %
	acikbilim.yok.gov.tr	

Dr. Lic. Mat. Walter Arriaga Delgado
DNI: 16732082
Asesor

18	Fuente de Internet	<1 %
19	id.123dok.com Fuente de Internet	<1 %
20	res.mdpi.com Fuente de Internet	<1 %
21	www.numerade.com Fuente de Internet	<1 %
22	Submitted to University of Duhok Trabajo del estudiante	<1 %
23	Cai, W.. "An Adaptive Spline Wavelet ADI (SW-ADI) Method for Two-Dimensional Reaction-Diffusion Equations", Journal of Computational Physics, 19980101 Publicación	<1 %
24	Submitted to Info Myanmar College Trabajo del estudiante	<1 %
25	Submitted to University of Wales Institute, Cardiff Trabajo del estudiante	<1 %

Excluir citas	Activo	Excluir coincidencias	< 15 words
Excluir bibliografía	Activo		

Dr. Lic. Mat. Walter Arriaga Delgado
DNI: 16732082
Asesor

RECIBO DIGITAL

turnitin

Recibo digital

Este recibo confirma quesu trabajo ha sido recibido por Turnitin. A continuación podrá ver la información del recibo con respecto a su entrega.

La primera página de tus entregas se muestra abajo.

Autor de la entrega:	Eduardo Oliva Suarez
Título del ejercicio:	Informe final 01
Título de la entrega:	Final-final
Nombre del archivo:	tesis-EOlivaS.pdf
Tamaño del archivo:	1.48M
Total páginas:	69
Total de palabras:	16,967
Total de caracteres:	72,479
Fecha de entrega:	24-feb.-2023 05:14p. m. (UTC-0500)
Identificador de la entre...	2022350880

"Localización y corrección de errores en la transmisión de mensajes mediante código BCH"

Dr. Lic. Mat. Walter Arriaga Delgado
DNI: 16732082
Asesor

UNIVERSIDAD NACIONAL“PEDRO RUIZ GALLO”
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
ESCUELA PROFESIONAL DE MATEMÁTICA

Localización y corrección de errores en la transmisión de mensajes mediante códigos BCH

Tesis

Para optar el Título Profesional de

Licenciado en Matemáticas

presentado por:

Bach. Mat. Eduardo Enrique Oliva Suárez

Asesor

Dr. Lic. Mat. Walter Arriaga Delgado
ORCID: 0000-0001-9311-5314

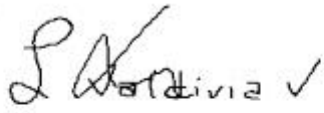
Lambayeque – Perú

Marzo – 2023

Universidad Nacional Pedro Ruiz Gallo

Escuela Profesional De Matemática

Los firmantes, por la presente certifican que han leído y recomiendan a la Facultad de Ciencias Físicas y Matemáticas la aceptación de la tesis titulada “Localización y corrección de errores en la transmisión de mensajes mediante códigos BCH”, presentado por el Bach. Mat. Eduardo Enrique Oliva Suárez en el cumplimiento parcial de los requisitos necesarios para la obtención del título profesional de Licenciado en Matemáticas.



Dr. Lic. Mat. Segundo Leonardo Valdivia Velásquez

Jurado Presidente

Dr. Lic. Mat. Rubén Esteban Burga Barboza

Jurado Secretario

Lic. Mat. Miguel Angel Baca Ferreyros

Jurado Vocal

Fecha de defensa

Marzo – 2023

Localización y corrección de errores en la transmisión de mensajes mediante códigos BCH

Oliva Suárez Eduardo Enrique

Resumen

El presente informe de investigación titulado "Localización y Corrección de errores en la transmisión de mensajes mediante códigos BCH", tuvo como objetivo principal localizar y corregir errores en la transmisión de mensajes haciendo uso de los códigos BCH, producidos por disturbios a causa del ruido mejorando en su recepción de los mismos. La metodología de estudio fue del tipo Inductivo – Deductivo, el modo de la investigación fue Unidisciplinario, el tipo de investigación que se usó fue Científica teórica y aplicativa. Se utilizó un conjunto de palabras código de una cierta longitud pertenecientes a un campo binario y haciendo el uso de polinomios se encontró una matriz generadora usando cambios cíclicos de un código cíclico, además se aplicó este proceso en el grupo finito o campos de Galois $GF(2^r)$, donde se encontró una matriz generadora llamada *matriz de verificación de paridad H 15x8* de un código cíclico de longitud 15, llamado código 15 o *C15,* pertenecientes a la familia de códigos BCH, la cual se utilizó esta matriz para codificar palabras pertenecientes al *C15* para el proceso de transmisión y para la decodificación se utilizó un algoritmo de decodificación la cual se hizo uso del polinomio síndrome y la matriz de paridad para encontrar los posibles errores en la transmisión y a través de una ecuación cuadrática se utilizó sus raíces que pertenecen al $GF(2^4)$ para corregir los posibles errores producidos en la transmisión. Por tal razón se localizó y corrigió errores en la transmisión de mensajes haciendo uso de los códigos BCH, siendo una clase de códigos de corrección de errores dobles la cual nos permite la mejora de envío y recepción de información, recomendando que esta práctica sea aplicada en el campo de las telecomunicaciones e informática, con el objetivo de mejorar la trasmisión de mensajes detectando y corrigiendo errores, producidos a causa del ruido.

Palabras Claves: códigos, mensajes, transmisión, localizar y corregir

Abstract

The present research report entitled "Location and Correction of errors in the transmission of messages through BCH codes", had as main objective to locate and correct errors in the transmission of messages using the BCH codes, produced by disturbances due to noise, improving upon receipt thereof. The study methodology was of the Inductive - Deductive type, the research mode was Unidisciplinary, the type of research that was used was Theoretical and applied Scientific. A set of code words of a certain length belonging to a binary field was used and using polynomials, a generating matrix was found using cyclic changes of a cyclic code, in addition this process was applied to the finite group or Galois fields GF(2^r), where a generator matrix called the H 15x8 *parity verification matrix* of a cyclic code of length 15, called code 15 or *C15*, belonging to the BCH code family, was found, which this matrix was used to encode words belonging to the *C15* for the transmission process and for the decoding, a decoding algorithm was used, which used the polynomial syndrome and the parity matrix to find the possible errors in the transmission and through a quadratic equation, its roots that belong to the GF(2^4) to correct possible errors produced in the transmission. For this reason, errors in the transmission of messages were located and corrected using the BCH codes, being a class of double error correction codes which allows us to improve the sending and receiving of information, recommending that this practice be applied in the field of telecommunications and computing, with the aim of improving the transmission of messages by detecting and correcting errors caused by noise.

Key words: codes, messages, transmission, locate and correct